Procedural Reconstruction of Architectural Parametric Models from Airborne and Ground Laser Scans

Edum-Fotwe, Kwamina

Award date:
2018

Awarding institution:
University of Bath

Procedural Reconstruction of Architectural Parametric Models from Airborne and Ground Laser Scans

Kwamina Edum-Fotwe

A thesis submitted for the degree of
Doctor of Engineering

University of Bath
Department of Computer-Science

September 2017

Table of Contents

Thank You

Paul, Matt, Richard and Dan

Yongliang, Phil and Darren

Abena, Mawunyo, Emefa, Mawuena and Esther

Danny, Chris, Christopher, David, James, Amos, Marius, Annatt, Joel, Tony, Peter, Brad, Triveni, Christos, Amy, Kelly, Simon, Nick, Nathan, Damien, Amanda, Steve, Christian and Nikos

Hashim, Dan, Matt, Milto, Dhana, Alastair, Alex, Anna-Marie, Max and Ben

Denise, Anna, Zoe, Brent, Peter, Sarah and Carol

Joe, Joey, Sarah, Richard, Tom, Marc, Chrissy, Mav, Keith and Andy

Mother and Father

Abstract

This research addresses the problem of efficiently and robustly reconstructing semantically-rich 3D architectural models from laser-scanned point-clouds. It first covers the pre-existing literature and industrial developments in active-sensing, 3D reconstruction of the built-environment and procedural modelling. It then documents a number of novel contributions to the classical problems of **change-detection** between temporally varying multi-modal geometric representations and **automatic 3D asset creation** from airborne and ground point-clouds of buildings. Finally this thesis outlines on-going research and avenues for continued investigation - most notably fully automatic temporal update and revision management for city-scale CAD models via data-driven procedural modelling from point-clouds. In short this thesis documents the outcomes of a research project whose primary aim was to engineer fast, accurate and sparse building reconstruction algorithms.

Formally: this thesis puts forward the hypothesis (and advocates) that architectural reconstruction from actively-sensed point-clouds can be addressed more efficiently and affording greater control (over the geometric results) - via deterministic procedurally-driven analysis and optimisation than via stochastic sampling.

List of Abbreviations

This section summarises the mathematical notation and acronyms used in this document.

Mathematical Notation

$\cup$ - union, $\cap$ - intersection, $a - b$ - difference (a minus/subtract b), $\oplus$ - exclusive disjunction (xor), $\sum$ - summation, Δ - derivative, $\int$ - integral, $|a|$ - cardinality or length, $||a||$ - magnitude, $\forall$ - for all, $\in$ - in, ! - negate or not, $\equiv$ - equivalent, $\not\equiv$ - not equivalent, $\approx$ - approximately equal, $\leftarrow$ - gets, $\rightarrow$ - to

Acronyms and Shorthands

LiDAR - Light Detection and Ranging
ToF - Time of Flight
SAR - Synthetic Aperture Ranging
SLAM - Simultaneous Localisation and Mapping
MVS - Multi-View Stereo
SfM - Structure from Motion
IoU - Intersection over Union
DoF - Degrees of Freedom
R2 - Real Valued 2-Dimensions
R3 - Real Valued 3-Dimensions
R4 - Real Valued 4-Dimensions
RN - Real Valued N-Dimensions
DEM - Digital Elevation Model
DSM - Digital Surface Model
DTM - Digital Terrain Model
nDSM - Normalised DSM
DoEM - Difference of Elevation Models
ESRI - Environmental Systems Research Institute
CAD - Computer Aided Design
CSG - Constructive Solid Geometry
CAG - Constructive Area Geometry
BIM - Building Information Modelling
AABB - Axis Aligned Bounding Box
KD - K Dimensional
ND - N Dimensional
PLY - Stanford University Polygon File Format
OFF - Princeton University Object File Format
OBJ - Wavefront Geometry Definition File Format
STL - StereoLithography Format
GML - Generative Modelling Language
RIB - Renderman Bytestream Interface
DXF - AutoCAD Drawing Interchange Format
CSV - Comma Separated Values
PCD - Point-Cloud Data (PCL File Format)
PTS - Unstructured Point-Cloud Format
PTX - Unstructured Point-Cloud Format
XYZ - Unstructured Point-Cloud Format
RMS - Root (of the) Mean Squared
SSD - Sum (of the) Squared Differences
FFT - Fast Fourier Transform
DT - Delaunay Triangulation
CDT - Constrained Delaunay Triangulation
CCDT - Conformal Constrained DT
SLS - Straight Line Skeleton
SIFT - Scale Invariant Feature Transform
MSER - Maximally Stable Extremal Regions
MAT - Medial Axis Transform
SVM - Support Vector Machine
NN - Neural Network
MST - Minimum Spanning Tree
GC - Generalised Cylinder
BSP - Binary Space Partitioning
SDF - Signed Distance Field
DoG - Difference of Guassian
DoN - Difference of Normals
PCA - Principal Component Analysis
SVD - Singular Value Decomposition
LLS - Linear Least Squares
MLS - Moving Least Squares
ICP - Iterative Closest Point
IoP - Intersection of Planes
RANSAC - Random Sampled Consensus
w.r.t. - with respect to
s.t. - such that
IO - Input Output
GL - (Open) Graphics Language
GLSL - OpenGL Shading Language
PG - Procedural Generation
CG - Computer Graphics
CV - Computer Vision
UI - User Interface
GUI - Graphical User Interface
RGB - Red Green Blue
RGBA - Red Green Blue Alpha (Transparency)
ARGB - Alpha (Transparency) Red Green Blue
RGB-D - Red Green Blue Depth
HSV - Hue Saturation Value
HSB - Hue Saturation Brightness

During the course of this thesis - a number of new acronyms are introduced to denote concepts and novel algorithmic operators. As a supplement to the pre-existing abbreviations - this section documents the undefined shorthands that will be employed - as this document progresses - for reference.

DEV : Detector for Engineering Variance
MAMMAL : Maximal Area 2.5D Mass-Modelling of Airborne LiDAR
ARROW : Accurate Railed Reconstruction of Openings and Walls
CUBE : Complete Unified Building Exteriors
ACCRA : Automatic City-CAD Revision Agent

MARS : Maximal Area Roof-shape Segmentation
QUALM : Quick Unconstrained Approximate L-Shape Method
GRAILS : Graph-Refined Approximate Interior Linear Spine
SLADE : Split-Logic-Axes Detector and Extractor
DRAPES : DEM-Reconstruction for Accelerated
Partitioning of Engineering Surfaces

LBET : Linear Boundary Edge Traversal
LIET : Linear Internal Edge Traversal
MAMP : Maximal-Area $\rightarrow$ Minimum-Primitives
PIP : Piecewise Intersection of Planes

The first block of acronyms corresponds to the principal algorithms that are defined and exploited in this research - the second block to auxiliary and supporting geometric operators and the third to useful ideological concepts.

Chapter 0

Introduction

This thesis presents a thorough treaty of the state of data-driven procedural modelling for architectural visualisation and simulation and prescribes a number of reconstructive algorithms to advance the state of the art - which efficiently extract sparse architectural geometry from airborne and terrestrial laser-scanned point-clouds - in a manner suited to interactive rendering.

Over the last two decades, the explosion in the use of actively-sensed point-clouds has been without relent. From industrial applications (in surveying and environmental analysis) right through to consumer level devices (such as Microsoft's Kinect), the use-cases for acquiring 3D depth information of the real-world seem set to continue to grow. Point-clouds are simply collections of points in space. They are typically acquired by the use of Light Detection and Ranging (LiDAR) scanners - which consider the return rate of a laser beam in order to determine the distance from the sensor to a surface. However there are a multitude of other ways to create and acquire point cloud data - ranging from photogrammetry (or image-inversion), magnetic resonance imaging, mesh sampling, and procedural synthesis.

Each point in a point-cloud describes the location of a position on a surface - typically using a 3D cartesian coordinate system [x,y,z]. However increasingly points can also embody additional information such as the derivative of the gradient of change of the surface at the position (the surface-normal) and the colour of the surface at the position (if the sensor combines depth with an RGB stream). As a result active-sensing is a highly utilised component of computational systems that perceive the physical world.

Point clouds have many benefits over other sensing mediums - such as photographs. Since the energy used to measure distance is actively supplied the magnitude of the signal-error is typically far smaller than with passive sensors. Fundamentally though the key difference is the ability to quantify the geometric error present in models recovered from actively sensed point-clouds, relative to photogrammetric models. Point clouds provide one of the simplest discrete representations of geometric form. As each point is distinct (free of topology) they are well suited to being efficiently rendered - which enables volumous amounts of data to be visualised in 3D in real-time.

Unfortunately point-clouds are not perfect and a number of key problems have plagued their exploitation since the advent of the field of active sensing. The

key challenges in processing laser-scan data are commonly known to be feature-detection, segmentation and approximation. At a high-level the challenge in each is a product of the fact they are ill-defined problems.

Though great strides have been made in surface-reconstruction (a dense form of approximation), there is typically little semantic information embedded in a reconstructed surface and additional processing is required to turn the reconstructed geometry into an effective digital asset. This can be a costly-process at scale, and can negate the benefits of automation if it requires semi-automation or manual human intervention. Further in this regard, within the domains of architectural modelling, visualisation and simulation there are a number of conventions that preclude the use of un-refactored dense surface reconstructions. Foremost the demand for topological control in the construction of geometric assets. However further the requirements for succinct, semantically meaningful models that clearly communicate the function and form of each architectural component. Although the gap between manually-constructed and automated models is still quite large a number of vital contributions have taken great strides in bridging the gulf.

This thesis aims to further close this gap, and presents a set of simple, automatic algorithms designed to efficiently yield clean, compact, semantically rich architectural models from various types of laser-scanned point-cloud.

The key idea, and the goal of the project is enabling Automatic Temporal Updates to City-Scale Urban CAD Models driven by Multi-Modal Laser-Scans. In essence the aim is to be able to automatically maintain up-to-date city-models in rapidly changing (constantly evolving) urban environments by analysing, segmenting and modelling high-quality, 3D building models in aerial and ground LiDAR. Additionally the project seeks to ensure that the generated geometry is both accurate enough for use in surveying and analytic tasks, and compact enough to be used in real-time within interactive 3D simulations and environments.

Document Overview

This document is divided into two parts : Research and Development. The first part (Research) analyses the pre-existing techniques within the domain, considering the strengths and detailing the recurrent problems. The second part (Development) presents and analyses the novel contributions.

In order to progress the current state of the art, this thesis addresses three vital problems in architectural reconstruction. These vital problems are:

1. Understanding Temporal Geometric Changes
2. Automatically Modelling Architecture in Airborne Laser Scans
3. Automatically Modelling Architecture in Ground Laser Scans

Each development chapter presents an intuitive approach to addressing the corresponding problem. The final development chapter discusses an automatic agent that amalgamates the key novel algorithms developed in pursuit of an artificially intelligent architectural CAD technician :-

The Procedural Revision Agent.

Key Dichotomies

Before delving into the meat of this thesis, this section briefly introduces the key dichotomies that are often considered within the domain of architectural reconstruction. These contrasting adjectives represent the key operational and behavioural trade-offs that exist between representations and techniques within the field.

Note: the formal definitions that accompany each of the following pairs of contrasting terms are drawn from the Oxford Dictionary of English. [29]

Sparse vs Dense (*thinly dispersed or scattered* versus *closely compacted in substance*). Here the term sparse refers to compressed (or lightweight) representations whilst dense refers to un-compressed (heavy-weight) representations. Sparse representations are generally better at minimising the computational requirements for further geometric processing, however dense representations are generally more accurate (since they are explicit and not simplifications). The important thing, is that any decision as to which class of algorithm is preferable (those that return sparse models relative to those that yield dense models) is heavily tied to the context of the resultant use of the outcome models. In essence, whether a sparse or dense reconstruction is better depends entirely upon what an end-user intends to do with a reconstruction. In the case of city-scale simulation, sparse methods are preferable, whilst for analysis of a single building dense methods may be preferable. However critically although these two attributes contrast, in reality the classification of an algorithm as being sparse or dense is often not a binary distinction, but rather a sliding scale between these two extremes.

Discrete vs Continuous (*individually separate and distinct* versus *forming an unbroken whole; without interruption*). The key difference between discrete and continuous representations is analogous to the difference between fixed resolution and multi-resolution 2D graphic representations. As an example a photograph or image is discrete whilst a scalable vector graphic is continuous (since it can be resampled at arbitrary resolutions without the loss of information). Further one could consider a point-cloud a discrete representation and a tetrahedra-mesh a continuous representation. However from another perspective a tetrahedra-mesh could be considered discrete whilst an algebraically defined signed distance field could be considered a continuous representation. This simply demonstrates that the association of a type of representation as either discrete or continuous is often relative to an alternate representation type. The key however, is that fundamentally there are only two types of abstract transformation that can be applied to alter the continuity of a geometric representation. Either moving from a discrete representation to a continuous one (as is common in active sensing and 3D reconstruction) or the inverse - moving from a continuous representation to a discrete one (as is the case in 3D and 2D rendering algorithms).

Explicit vs Implicit (*stated clearly and in detail - leaving no room for confusion or doubt* versus *suggested but not directly expressed - (of a function) not expressed directly in terms of independent variables*). Geometrically this refers to the manner of shape representation. An explicit representation such as a point-cloud states unambiguously the exact position and attributes of each point. However an implicit representation such as a CSG-tree implies the structure of a solid-object in terms of a hierarchy of boolean operations without directly specifying the actual position

of its surface. Implicit representations encode the key attributes of an entity in an abstracted manner.

Accuracy vs Brevity (*the quality of state of being correct or precise* versus *concise - brief but comprehensive - close to the actual, but not completely accurate or exact*). Although in the strictest sense accuracy and brevity are not antonyms, in practice there is generally a trade-off between how accurate a geometric representation is and the brevity (or level of compression) it exhibits - often the greater the accuracy the less the compactification, and conversely the greater the conciseness the less the accuracy. However this is not a hard rule, and recently the academic trend has been investigation of methods that are as sparse as possible whilst meeting an error tolerance.

Deterministic vs Stochastic (*the doctrine that events and operations are ultimately controlled (determined) by explicit causes as opposed to chance* versus *having a random probability distribution or pattern that may be analysed statistically (for example probabilistically) but may not be predicted precisely*). The key distinction here is between operators whose behaviour is governed (controlled) by logic or reason, against operators that rely upon (exploit) chance. Although one could argue that stochastic processes can be implemented such that they perform deterministically (through the use of seeded/pseudo-random number generators), the core difference is ideological. Fundamentally random sampling strategies cannot be trivially controlled or profiled and whilst such stochastic methods are becoming prevalent in Computer Vision (wherein there is generally little or no intrinsic relationship between pixel colour and an underlying surface's geometry), within Computational Geometry the preference is for deterministic methods whose return values are solely products of the input. Indeed determinism is a fundamental building block of Computer-Science without which many integral pursuits such as formal proof of a program's correctness would not be possible. Despite this stochastic methods tend to find use within domains for which deterministic solutions to problems are either too complex to implement or computationally expensive - for example in realtime applications.

Procedural vs Manual (*defined, operated or controlled algorithmically - or autonomously* versus *operated or controlled by hand, rather than automatically)*. These terms can be applied to both operations and representations. Operationally good synonyms for these terms are automatic versus interactive. In terms of representations, one could use the loose analog implicit versus explicit. Essentially - whenever the term *procedural* is used it simply refers to functional, procedure-driven or algorithmic phenomena.

Structured vs Unstructured (*of the relations between parts or elements of something - arranged according to a plan - possessing a pattern or organisation* versus *without (lacking) formal organisation or structure*). This relates to the manner in which points in a point-cloud are arranged. They may possess a consistent uniform arrangement (as in gridded structured parallax range-images) or they may be scattered arbitrarily without consistent inter-point spacing (as in unstructured fixed-position radial range-scans).

Each of these juxtapositions outlines a key analytic, operational or representa-

tional contrast. They are referred to through-out this thesis in order to clarify the behaviour of both the pre-existing and newly-proposed methods. It is important to understand that ultimately there are no single absolute *correct* solutions to the key problems addressed by this research. However these terms aid quantitative and qualitative discussion and evaluation. They provide a base vocabulary that is useful in categorising the behaviour of different geometric operators and representations within the domain.

Notes on Document Presentation

This section outlines the presentational conventions employed by this Thesis.

Use of Colour: is an important component of the visual presentation of this document. Throughout the development chapters colour figures are used to clarify important concepts and results. It is therefore recommended that this document be viewed in colour. Note: additionally that this document is designed to be viewed digitally. Whilst some readers may prefer a printed colour version - be aware that facilities such as the interactive index and internal document links for references in the bibliography shall be lost when printed.

Novelty Flags: these are short descriptive statements that draw attention to the novel aspects of each method - and are included to help readers quickly identify the important benefits and contributions. Each novelty flag is denoted by green typeface. The primary aim of this form of mark-up is to make it easy to see (at a glance) statements and ideas that relate to fresh contributions to knowledge and/or improved algorithmic performance relative to pre-existing techniques.

0.1 Industrial Context and Motivations

This section provides background on the industrial context of this research and its underlying motivations. It introduces the industrial sponsor, their requirements and the core problem that they face which prompted these investigations.

Cityscape Digital is an architectural visualisation firm based in London. Cityscape produce high-fidelity graphical representations of the built environment - to support various stages of the development life-cycle within architecture and construction. From pre-planning, through design, feasibility and build, Cityscape's still, animated and interactive visualisations enable exploration, verification, communication and marketing of both large and small scale urban developments.

Cityscape (as many digital-media firms) rely heavily on 3D CAD models. Indeed 3D geometric representations of the physical world are a fundamental pre-requisite to many of their industrial activities. However the rising cost of acquiring and maintaining such assets poses many challenges for the company. In particular - as the scale and scope of the projects Cityscape engages in grows - so to does the requirement to reduce the turn-around time for producing and maintaining semantically rich, up-to-date 3D building models.

Note: within this context semantically rich simply means 3D models that carry (embed) semantic meaning. Essentially the structure of the 3D representation enables manipulations and operations (i.e. editing and rendering) and analysis at the level of semantic components (such as roof, walls, windows and doors) as opposed to at the level of lower-level geometric elements (such as vertex, edge, face).

Presently Cityscape acquires their 3D building models through a combination of off-the-shelf providers (such as Vertex-Modelling and Z-Mapping) and in-house modelling. The main benefit of buying pre-made models from such geo-spatial providers is that it can reduce the production turn-around time - however the cost of such purchases can be quite high - often in the range of several thousand pounds per-project. Cityscape have observed that that in some instances this expense can be prohibitive as it eats into a project's budget and reduces the resources available to spend on in-house development. Additionally Cityscape have noted that there is typically no guarantee of the absolute accuracy of these assets and as such off-the-shelf models often require further editing prior to being used within the digital-content produced by the company. These artefacts and anomalies can be the product of temporal disparity (the fact the purchased models are no longer up-to-date) and sporadic modelling errors (such as over-simplifications and incorrectly modelled photogrammetrically derived components). To make matters worse - even when the geometric form of these off-the-shelf models is fundamentally correct - there can be technical issues - such as inconsistently oriented facets that interfere with processes such as texturing and rendering.

Furthermore (and likely the greatest limitation of relying heavily upon external suppliers is that) there is no guarantee that a geo-spatial data provider will possess adequate level-of-detail models for a region of interest - which means that the company cannot always rely solely on buying in baseline models.

On the other hand - manually producing such pre-requisite assets in-house (though incurring a lower initial economic investment) - demands a substantial investment of time - and puts a greater strain on Cityscape's modelling team as it reduces the time they have to spend on higher-level modelling tasks - forcing them to focus on the construction of boiler-plate geometry.

Therefore (from Cityscape's perspective) the rationale behind this research is simple. They seek to simultaneously reduce the cost of production in terms of the economic expense and the turn-around time for their *raw-material* - so as to increase their per-project profit margin and the scale of developments they can support.

Prior to this research Cityscape trialled a number of systems to automate the process of urban modelling from various types of sampled data - most notably sets of photographs and point-clouds. However they are yet to identify an effective solution that meets their needs. Cityscape have noted that the key limitation of the reconstructive methods they have employed is the quality of the building models that are generated and the large (exponential growth in) runtimes for the automatic algorithms that process the data. In particular - in some of their experiments they found that automated reconstruction was only feasible as an offline task on specialist hardware. For example they have had to leave algorithms running over-night on the companies' cluster in order to generate results. Whilst in principle compute is cheap relative to human labour - in practice the power required to coordinate automatic building reconstruction eats into the resources afforded photo-real rendering - which is the companies main source of revenue.

Essentially Cityscape have observed that the reconstructive algorithms that create high-quality models scale poorly to city-scale datasets whilst the algorithms that execute quickly typically yield low-fidelity geometry that is unsuitable for their use-case. Now whilst this problem is not unique to Cityscape it has become apparent to the company that they have reached somewhat of bottleneck for which their continued growth and prosperity is tied to their ability to resolve this problem.

0.2 Research Methodology

This section outlines the research methodology that pervades these investigations. The aim is to provide readers with a deeper appreciation of the underlying approach to research that is employed - beyond the narrower cyclical process of analysis, experimentation, implementation and performance optimisation.

The main point is that these investigations call for both high-level ideological advancements and low-level computational improvements to the way architectural models are recovered from laser-scanned points clouds. In order to ensure the material is accessible to readers from various disciplines - this thesis focusses on exposing both the technical developments and the observations and insights that enable them. To put it another way - this thesis documents the outcome of an iterative process that involves: first analysing the pre-existing literature to determine points of weakness and the major limitations - then experimenting with operators and algorithms that can combat the problems identified - then implementing and testing the most appropriate options and then (once viability is determined) opti-

mising performance using software-engineering techniques. Due to the nature of the problem at hand - i.e. the fact that high-quality building reconstruction could be considered an ill-defined (or under-constrained) task - this process is in many ways open ended. In this sense the work presented documents the state of development at the point that the funded period of the project drew to an end. As such it is important to be aware that this work represents the progress to date (so to speak). This thesis does not claim to have solved all the problems that exist within the domain. It simply documents improvements developed during the course of these investigations. Whilst some of these improvements represent significant enhancements to the state of the art - others are incremental in nature.

Further more in terms of the critical evaluative measures (that are of most concern) in these investigations - this research focusses on addressing the quantitative attributes of building reconstruction methods more so than their qualitative attributes. Do not misunderstand - the aesthetic appearance of the CAD models generated by the prescribed techniques is of vital importance (especially to parties such as Cityscape). However this research takes the view that ultimately - it is easier (and far more practical) to measure algorithmic gains quantitively than qualitatively. In other words the focus of the evaluative effort is in determining the geometric accuracy, growth in execution time and geometric compactification (of the assets generated) as opposed to subjective measures such as an end-user's perception. Interesting though, this work demonstrates that their is an inherent relationship between these quantitative metrics and derived qualitative metrics that govern how *good-looking* the reconstructed building models are.

0.3 Outline of Contributions and Impact

Before progressing, this section briefly summarises the key contributions to human knowledge derived from this research. These ideas and algorithms are covered in greater detail within the development chapters - however it is quite useful to be aware of the novelty of this work as a preliminary to analysis of the literature.

The Hypothesis - The Central Idea and The Main Conceptual Argument:
Deterministic (Non-Stochastic) Procedural Optimisation enables Efficient Sparse Reconstruction of High-Quality Building Models from Laser-Scanned Point-Clouds.

The Algorithms - The Primary Technical Developments:

DEV: Multi-Modal Semantic Change-Detection for Selective Reconstruction

MAMMAL: Fast, Accurate and Sparse, Automatic 2.5D Building Reconstruction

ARROW: Fast, Accurate and Sparse, Automatic 3D Facade Reconstruction

Note: the final chapter provides a fuller enumeration of the novel concepts and technical advancements developed and prescribed by this research.

Part I

Research

Chapter 1

Literature Review

This thesis addresses the problem of recovering semantically meaningful architectural models from laser-scanned point clouds. As part of this, three key problems (outlined in the introduction) are tackled. For reference these are:

1. Understanding Temporal Geometric Changes
2. Automatically Modelling Architecture in Airborne Laser Scans
3. Automatically Modelling Architecture in Ground Laser Scans

The driving aim of this project is to determine the feasibility of continuous geometric update to city-scale CAD models using laser-scanned point-cloud data. Before delving into the heart of these core problems this part of the thesis covers the pre-existing literature which informs the developments presented in part two.

This literature review is divided into two sections. The first section revises the key tasks of active-sensing with a focus on the computational aspects of point-cloud processing for architectural reconstruction. The second section covers the domain of procedural modelling, focussing on the algorithmic definition and manipulation of 3D building models. This literature review then concludes with a summary of the important contributions and identified recurrent sub-problems.

1.1 Architectural Reconstruction

This section reviews pre-existing architectural reconstruction methods.

It is organised as follows:

- Primer: Categorising Building Reconstruction Methods
- Timeline of Contributions and Developments
- Key Related Work: Case Studies
- Limitations and Recurrent Problems

1.1.1 Primer: Categorising Building Reconstruction Methods

This preliminary section outlines basic concepts in architectural reconstruction from laser-scanned point-clouds. It considers the characterising attributes of building reconstruction methods, alongside the heuristics and priors commonly applied to automatically produce 3D building models from sampled representations.

Automatic vs Interactive

A building reconstruction method can be considered automatic or interactive. Automatic methods do not require a human in the loop whilst interactive methods do.

LiDAR-based vs Image-based

Further more one can consider the type of input data to a reconstructive method as a means to categorise them. LiDAR-based (laser-scanning) strategies operate on actively sensed point-clouds. Image-based (photogrammetric) methods operate on sets of photographs or sparse MVS points. Note: there are also hybrid methods that employ both laser-scanned point clouds and image data.

Data-Driven vs Model Driven

Another key attribute that distinguishes building reconstructive methods from one another is whether they are *data-driven* or *model-driven*.

Data-driven methods are predicated on recovering boundary representations based solely on the data-provided. In this sense they are direct methods.

Whereas model-driven (or library-based) methods rely on a pre-existing *knowledge-base* (or model-library) in order to recover boundary representations that conform to a pre-determined abstract *model* of what constitutes a building. As such model-based approaches can be thought of as in-direct.

Shortly this thesis considers key examples of data-driven and model-driven building reconstruction methods. However at a high-level it is useful to be aware that data-driven reconstruction methods are generally favourable for their efficiency and geometric accuracy - whilst model-based reconstructive methods are often favourable for their ability to mitigate sensing noise and their ability to produce higher-quality boundary representations.

Analytic Methods vs Stochastic Methods

Further more in terms of dichotomies applicable to building reconstruction methods one can also consider an algorithm as analytical or stochastic.

Analytic methods (as their name suggests) rely upon analysing input point-clouds and exploiting geometric predicates and constructions in order to localise upon and model boundary representations of architectural features.

Stochastic methods however, rely upon randomly sampling subsets of input point-clouds in order to localise on architectural features by exploiting chance in a manner akin to *trial-and-error* (or probabilistic modelling).

In essence, analytic methods resolve algorithmic decisions automatically by explicitly *computing* intermediary distinctive attributes and properties of a sampled-set, whilst stochastic methods resolve algorithmic decisions by *sampling* (or guessing) randomly until good approximates are identified. Further more, if an reconstructive method's result is solely a product of the supplied input (i.e. it is predetermined),

one can consider it *deterministic*.

As mentioned in the introduction - generally within image-based building reconstruction (photogrammetry) stochastic techniques are employed - since there is often little intrinsic relationship between pixel-intensity and a desired output model. However in geometric problems the desire is generally for determinisitic solutions to problems. Despite this, an alarming number of methods ([1], [67], [142], [73]) for building reconstruction from actively sensed point-clouds have arisen of late that employ stochastic techniques in order to forgo the difficultly associated with developing effective efficient analytic algorithms. On one hand, this is somewhat understandable given the seeming complexity of traversing large unordered sets of points. However (particularly in architectural reconstruction) the key benefit of effective general analytic methods is efficiency relative to stochastic methods. The key challenge though in the formulation of analytic reconstructive methods is generally the demand for domain specific insight. Indeed the efficacy of an analytic method rests largely upon the robustness (versatility, generality, flexibility) of the insights employed. Conversely, stochastic methods are easy to write, yet harder to optimise in terms of computational efficiency.

Having outlined the types of building reconstruction algorithm that exist, the next subsection briefly outlines common priors that are employed.

Architectural Priors

Certain geometric structural priors are frequently employed by building reconstruction algorithms in order to control the nature of boundary-representations that are constructed. The key aim of embedding such priors in an algorithm is to constrain the 3D models that are produced and preserve and/or enforce desirable features and attributes. These features and attributes include: Planarity, Smoothness, Continuity [107], Parallelism, Orthogonality (Right-Angles) [166], Symmetry - Reflectional, Rotational [166], Principal Directions, Common Angles [166], Manhattan-Worth Assumption [143][163],[106] and Sharp/Crease Edges/Ridges [165].

Note: the exploitation of geometric priors and constraints represents one of the key differences between architectural reconstruction algorithms and general purpose surface reconstruction algorithms. The key idea is that the built-environment exhibits common regularities that are not typically present in scans of arbitrary objects. Algorithms can reason about building representations in a manner that is not amenable to general surfaces - due to the fact that 1) architecture is man-made (which means that the rules that govern it are not empirically derived but rather decisions made by humans), and 2) because the function of buildings (as dwellings for human-beings) significantly constrains their geometric form. Buildings may appear high-varied and distinct, however they all share common attributes (entrances, apertures, walls, ground-contact...) which are well-known and bound by the requirement to be structurally sound. However this cannot be generally said for organic objects or arbitrary man-made objects as examples.

Ultimately embedding one's pre-existing knowledge and expectations - about the types of geometric feature prevalent in architecture - enables algorithmic enhancements in terms of computational efficiency and model quality.

However the downside to exploiting priors is the *limiting* effect they can have on an algorithm's versatility in the presence of highly irregular buildings.

1.1.2 Timeline of Contributions and Developments

This timeline of architectural reconstruction steps through the last two and half decades reviewing the key academic contributions to the domain. Note: whilst the focus of this research is architectural reconstruction from LiDAR point-clouds this timeline also covers photogrammetric and hybrid approaches so as to provide a fuller account of prior techniques and developments.

Pre Millennium

Debevec's doctoral thesis [26] documents pioneering research into photo-real architectural modelling and rendering - from sets of photographs using a hybrid (geometric and image-based) approach. Debevec's semi-automated system (Facade) provides an end-to-end pipeline for the production of high-quality textured 3D building models that capitalises on the complimentary strengths of interactive 3D modelling and photogrametric stereo correspondence. By combining photographs with sparse geometry Debevec constructs far more compelling (life-like) 3D architectural representations than would otherwise be possible using manually created texture maps.

Lin and Nevatia [74] propose a monocular method for extracting rectilinear blocks to describe buildings from oblique aerial photography that relies on hypothesising and filtering roof components via a grouping methodology. However due to the nature of the input data (single-channel/grayscale images) the accuracy is limited - and they rely on an interactive editing tool to correct any errors.

Gulch, Muller and Labe [48] tend to the problem of multi-view photogrammetric 3D building modelling and in-particular automating the process of measuring buildings dimensions from image data to support creation of simple parametric primitives. Positively this approach supports the recovery of complex polygon shapes (i.e. that contain holes) and composite structures using CSG trees - however the accuracy of the approximate masses is contingent on user intervention.

Mayer [85] provides a survey and review of the photogrammetric building extraction techniques for aerial imagery prior to the turn of the millennium.

Brenner [10] reviews a suite of interactive modeling tools that are applicable to 3D building reconstruction and proposes a variant that uses DSM data and 2D ground-plans instead of aerial imagery.

Morgan and Tempfli [91] address fully automatic building extraction from DSM data based on plane-fitting using least-squares adjustment - and evaluation of roof-shape adjacency. Their key contribution is a morphological filter for distinguishing terrain points from non-terrain points. However the method does not support non-linear features and there is no consideration of computational efficiency.

Suveg and Vosselman [132] tend to the problem of automatic top-down building modeling from aerial images and 2D GIS maps. They note the further requirement to automate the process of partitioning buildings.

Brenner [11] discusses advancements in automatic generation of city models and proposes a generalisation of the straight line skeleton to construct roof-shape topology from 2D building footprints.

2000 - 2005

Vosselman and Dijkman [144] prescribe a plane-based segmentation and reconstruction strategy for generating 3D building models from ground-plans (building footprints) and airborne laser scans. This is a highly influential method and one of the first to exploit the relationship between a footprint and mass-components to infer the structure of roof-shapes to good effect. They discuss two related modeling strategies that deal with (resepctively) high and low density scans. Positively their approach yields clean and compact building representations, however its generality is somewhat limited by the reliance on orthogonality to regularise the roofshape decomposition induced by each footprint.

Kraus and Pfeifer [62] deal with robust derivation of digital terrain models from surface models. Their approach calibrates the laser scan data to resolve discrepancies between adjacent strips - to facilitate removal of the non-terrain features and (subsequently) water-flow analysis for interpolation (through break line analysis). Although this work deals with terrain recovery (as opposed to building extraction) it lays the foundation for geo-morphologically robust ground approximation which facilitates the use of normalised DSMs.

Rottensteiner, Briese and Jansa [112][114][113] address airborne building modelling from DSM data through a two stage process: analysis (to separate building boundaries from terrain) and reconstruction (to create planar models for detected buildings). Positively this method is quite well generalised due to the bottom-up curvature-based segmentation techniques employed. However there is no consideration given to computational efficiency and the preliminary results they present are lacking in terms of structural aesthetic quality. To combat this they introduce image data in an attempt to improve the geometric quality of the reconstructed models. However whilst the polymorphic feature extractor they employ certainly improves some cases it does not resolve all insufficiencies and introduces ambiguity (since there are multiple mediums under consideration) that must be resolved and that increases the complexity of the technique.

Fruh and Zakhor [40][41] propose an automatic method of creating textured 3D city models from airborne and ground DSMs and images. Positively the method is flexible (generalised) enough to deal with complex urban scenes and handles registration and correction of multi-modal data well. The inclusion of texture mapping also distinguishes it from the previous techniques. The key aspect of this technique is the linear algorithmic complexity of its sub-components which means it scales well. However the quality of the surfaces generated are less satisfactory and this is evident when they are rendered without textures.

Sithole and Vosselman [126] deal with classification of structures in airborne scans of urban regions. The key feature that distinguishes their work is that it is designed to deal with features such as bridges and ramps. However they only address classification and segmentation and not model generation.

You, Hu, Neumann and Fox [157] are amongst the first to address the extraction of complex building structures using primitive based fitting strategies. This includes the use of cuboid, flat-roof, slope-roof, cylinder and sphere primitives as well as super-quadrics to describe higher-order surfaces. Positively their approach produces higher quality results than pre-existing techniques - however the computational efficiency of the fitting strategies employed is not addressed.

Matikainen, Hyyppa and Kaartinen [84] address change detection between sampled sensory data and pre-existing vector maps [83] using a two stage process: building detection (based on classifying and segmenting the DSM) and change detection (using spatially driven membership coverage). Positively the precision and recall of their detector is strong - however it does not provide descriptors of the nature of the variance between altered buildings - other than if they are enlarged or new.

Oda, Takano, Doihara and Shibasaki [98] address 3D building reconstruction from airborne DSM data using linear-edges extracted from an edge-image using the Hough-transform. Positively this method produces cleaner building representations and supports texture mapping for representing facade detail. However it is constrained to linear edges (i.e. it does not handle curvature).

Rottensteiner, Trinder, Clode and Kubik [111] apply the Dempster-Shafer theory for data fusion to the problem of airborne building reconstruction from scan and image data. Whilst this improves upon Rottensteiner's earlier work the multi-modal approach still exhibits structural inadequacies that are not present in alternative operators.

Girardeau-Montaut, Roux, Marc and Thibault [45] propose an efficient change detection operator for unstructured ground laser scans for monitoring building sites - wherein the key operative constraint is time - i.e. it is designed to be a fast point-to-point method that can be used in realtime. They employ oct-tree data-structures to accelerate point-processing and provide scope for fast preview or more precise comparison (via consideration of Hausdorff distances). Note though that their process (though largely automated) still relies on a human in the loop.

Syed, Dare and Jones [133] address top-down building modelling from DSM and image data by mining planar roof patches from the DSM and constructing simple polyhedral models via bi-linear surface interpolation. The main limitation of this work is that it does not account for curvature or generalise to complex masses.

2005 - 2010

Wang, Lodha and Helmbold [148][147] tackle the problem of vectorising 2D building footprints via a probabilistic Bayesian formulation. Their approach is designed to deal well with sensitivity at the boundary of buildings, but requires pre-classified LiDAR data. The work yields high quality looking vector-shapes by maximising posterior probabilities using linear optimisation and simulated annealing - however (as a result) the absolute accuracy of the footprints suffers relative to non-probabilistic methods.

Zhang, Yan and Chen [160] address automatic building boundary extraction from airborne scans based on piece-wise line segments derived by extracting dominant directions from each dense zig-zag building boundary. Positively this enables the method to preserve orthogonality in the case of buildings with two principal directions - however the result is less robust for oblique edges (or buildings with more than 2 principal directions - i.e. those that do not conform to the Manhattan world assumption).

Belton and Lichti [7] propose the use of covariance analysis to address classification and feature extraction in unstructured ground laser scans based on local features of points (neighbourhoods) to estimate principal curvature.

Rabbini, Heuvel and Vosselman [107] propose an intuitive segmentation strategy for unstructured point-clouds that essentially looks for smoothly connected areas in point-sets by fitting planes and approximating curvature. Their method operates locally (i.e. it relies on surface normals and point-connectivity). Positively their approach prevents over-segmentation (so constrains the number of segments that are generated - preventing bloat) however it is primarily designed for industrial point-clouds (rather than architecture) and it is still subject to under-segmentation.

Akbarzadeh, Frahm, Mordohai et al. [3] tend to the difficult problem of reconstructing geo-referenced 3D urban scenes (building facades, street furniture, vehicles..) from video sequences in realtime. They rely on external GPS and INS readings (to geo-reference the results) and a hierarchical KLT tracker (to track correspondences between consecutive video frames). However the dense nature of the surfaces generated coupled with inaccuracies that result from the bundle-adjustment and filtering of the camera-trajectory renders the result unsuitable for architectural surveying applications that demand precision.

Koch, Heyder and Weinacker [61] propose a tree detection operator for airborne range scans that (though primarily being designed to facilitate forest management) can be applied to tree-filtering in urban reconstruction tasks.

Dorninger and Notheggar [31] tend to the problem of roof-shape segmentation from high-density unstructured airborne scans. Vitally their segmentation algorithm performs initial point clustering in parameter space so as to reduce the overall time complexity. Further their sequential implementation means that the computation time is largely independent of the number of input points and grows (rather) as a product of the number of segments. Unlike alternative roof-segmentation strategies - this does not operate in 2.5D but full 3D. Conceptually this is a strong method - that produces clean building modelling results - however as many operators it is restricted to planar components.

Tarsha-Kurdi, Landes and Grussenmeyer [137] address building roof modelling from airborne scans using RANSAC (i.e. random trial and error). The main problems with this work are that it provides no guarantee as to the accuracy of the roof patches it extracts or even that it will reliably extract all roof-shapes - and it can only deal with planar segments. There is also little consideration given to the computational performance of their random sampling strategy as point density increases.

Lafarge's doctoral thesis [64] tends to the problem of top-down building model recovery from high-res. satellite images. His approach uses DEM data to extract quadrilateral 2D footprints which are arranged (under constraints) to form compound structures. The conversion from 2D to 3D is addressed with a library of roof-type models (les mono-plans, les multi-plans, les non-planaires). As such whilst the visual appearance of the 3D models is reasonable and the method is capable of representing curvature - the absolute accuracy of the models generated by the approach suffers relative to techniques based solely on scan data. In this sense, Lafarge's stochastic Bayesian strategy trades off precision for higher-quality model generation. Note: additionally Lafarge's method only supports linear walls - meaning that whilst roof patches can be curved - building footprints can not directly represent curved facades.

Bosche and Haas [9] propose a fully automated object recognition strategy for detecting 3D CAD objects in unstructured laser-scans of construction sites. The

method is notable for its robustness in the presence of occlusion. Their operator is designed to support construction progress tracking - but its application to reverse-engineering of as built building models is not dealt with.

Sampath and Shan [117] deal with building footprint extraction from airborne scans via a point-to-point extremal boundary tracing strategy based on revising the convex hull of a set of points followed by a regularisation step that cleans footprints. Note: their work relies largely on the assumption of two perpendicular dominant directions - which limits its utility to complex structures - such as inner yards - and it can not deal with non-linear boundaries.

Schnabel, Wahl and Klien [120] employ the long-standing RANSAC paradigm to detect planes, sphere, cylinders, cones and tori from unstructured point clouds. Their method is designed to scale well (relative to pre-existing RANSAC based techniques) and perform robustly in the presence of heavy noise and many outliers. Positively - although this is a random method, it generalises quite well to arbitrary man-made objects (beyond architectural representations), and it recovers a CAD representation composed solely of shape proxies - however it is not capable of representing all classes of surface using its shape proxies - meaning irregular structures can not be directly extracted.

Müller, Zeng, Wonka and Van Gool [93] address automatic facade modelling from ortho-rectified images via a semantic sub-division strategy that encodes each facade model's descriptor as a shape tree. This is a powerful and heavily influential approach however it relies on template matching to describe the geometry of windows and doors - which means that although the results have high-visual quality, the real world accuracy is harder to verify. Additionally the method does not provide support for non rectilinear window or door shapes - which means it can only model regular facades.

Berg, Grabler and Malik [8] tend to visual recognition of architectural components in general images - and in particular the challenging problem of automatically segmenting building images into roof, wall, window and door components. Their approach uses fixed-size patch based features to transition from a generic appearance model to an image specific appearance model. Positively the results are strong - however the work's main application is in similarity based semantic image search rather than geometric reconstruction.

Pollefeys, Nister, Frahm, Akbarzadeh, Mordohai et al. [103] build upon their earlier work - proposing a system for automatic 3D reconstruction in real-time from video streams. This is a more complete version of their earlier system.

Tarsha-Kurdi, Landes Grussenmeyer and Koehl [138] compare and analyse model-driven and data-driven approaches to building modeling from LiDAR data - and document the primary characteristics of both reconstructive paradigms.

Lafarge, Descombes, Zerubia and Deseillingny [66][67] extend Lafarge's earlier work and formalise the aspects of the RJMCMC sampler employed (a Reversible Jump Monte Carlo Markov Chain sampler).

Sugihara and Hayashi [128] propose a system for creating composite top-down massing models with varying roof structures from pre-existing 2D maps. Their main contribution is a building partitioning scheme which is used to decompose each footprint into massing regions - however it is designed primarily for orthogonal building arrangements (with two principle directions) and there is no consider-

ation given to curvature (for facades or roof-patches).

Cheng, Gong, Chen and Han [16] deal with edge-detection and extraction for building analysis from multi-modal airborne data. In particular they focus on improving the correctness of photogrammetric edge detection techniques via the analysis of LiDAR scans. Though accurate - this method only extracts straight-line segments - it cannot detect curvature.

Park and Lim [100] propose a simple method of producing textured top-down models from airborne image and scan data. However their approach relies on manual intervention in 2D vectorisation which limits its scalability.

Zhou and Neumann [163] propose a versatile algorithm for creating compact watertight building models from airborne LiDAR that vitally (unlike many of its predecessors) generalises to an arbitrary number of principal directions by automatically learning from the data (i.e. it does not make assumptions about angles between facades). They also employ an analytic (differential geometry based) vegetation filter which uses an unbalanced SVM. The quality of the results are high and indeed the only limitation of this method is that it is only designed to deal with flat-top planar constructs automatically and they rely on user-interaction to represent non-flat objects.

Ding, Lyngbaek and Zakhor [30] present an algorithm for merging oblique aerial images onto 3D surfaces derived from airborne range scans. Note: this method does not produce the 3D surfaces directly - it focusses on the problem of accurate registration for texture-mapping.

Sampath and Shan [118] address automatic top-down building modelling by segmenting airborne point-clouds using covariance analysis to isolate planes and breaklines. This is a data-driven method that employs a roof-shape adjacency matrix to intersect neighbouring planes (i.e. the piecewise intersection of planes). The approach produces clean looking results but is limited to planarity only - there is also little consideration given to efficiency and algorithmic scalability.

Matei, Sawhney et al. [81] propose a segmentation method for extracting building footprints and basic structures on roofs directly from lower-resolution sparse airborne scans. Their building orientation estimation algorithm largely resembles a modified version of the Hough-transform.

Zebedin, Bauer et al. [159] tend to top-down building modelling from sets of aerial photographs by fusing sparse line features (to delineate height discontinuities) with dense roof surface data - under the banner of a graph cuts global optimisation. Their work provides basic support for representing roof curvature (with surfaces of revolution) but uses linear arrangements for walls and facades.

Dorninger and Pfeifer [32] discuss an approach to 3D building extraction based on plane segmentation and regularisation. Positively the quality of the generated models is high, however it is another plane-only approach - (i.e. it assumes all buildings can be modelled solely by planes).

Xiao, Fang, Tan et al. [153] address facade model creation from sets of street-side images using a semi-automatic SfM pipeline. They consider the facade as a developable surface and use the composited images to decompose its structure into a DAG (directed acyclic graph) via a top-down then bottom-up recursive subdivision. In terms of priors they rely on bilateral symmetry and repeated patterns.

Positively their approach yields higher quality sparse facade models that typical MVS methods - however the reliance on user interaction to correct segmentation errors presents an issue in terms of the drive towards fully automated methods.

Zhou and Neumann [164] devise an out-of-core streaming approach to reconstructing top-down building models from very large airborne scans that extends their earlier work - enabling city-scale data sets to be processed using lower amounts of memory via a state propagation technique that prevents seams (that otherwise commonly result from tile based padded streaming techniques).

Poullis and You [104] tackle city-scale reconstruction from airborne scans via a data-driven statistical segmentation method (that does not make assumptions about the input and as such bears no data dependencies) and a generalised model creation step that relies on three (generally fixed) parameters. The key limitation of their approach is that it only handles flat roof arrangements and footprints composed of staight lines (i.e. it cannot represent curvature). Additionally the boundary vectors produced a more susceptible to perturbations (caused by noise) than data-driven strategies such as Zhou's.

Hohmann, Krispel, Havemann and Fellner [51] address the tricky problem of fitting shape grammars to point-clouds and images of building facades. They suggest an end to end workflow for converting street-level architectural data into compact rewriting expressions in order to encode a semantically rich representation of facades using GML's stack based grammar. Although the early results are promising the generality of the technique is its main limitation in the sense that it relies on template grammars for each type of supported facade (i.e. it cannot automatically adapt to new classes of facade).

Chen and Zakhor [15] deal with tree-detection in large urban scenes by a two-step process of segmenting (by edge and height continuity) and classifying airborne scans (with a random forest classifier). Unlike their earlier work this method does not rely on RGB imagery and achieves precision recall rates upwards of 95% for large American and European test datasets.

Wang and Neumann [146] tend to the ill-defined problem of aligning aerial images and range-scans. They devise the 3CS (3-Connected-Segments) feature detector as a means to extract more robust (distinctive) correspondences and employ a two-stage RANSAC strategy to reduce registration error. There approach is designed to work well whenever the proportion of inliers in the data is low.

Jochem, Hofle, Rutzinger and Pfeifer [54] deal with segmentation of building roof-shapes in airborne scans so as to facilitate solar-gain calculation. There method takes into account the effect of shadowing (occlusion) caused by nearby objects (derived from the scan) - however it is only suited to roof-configurations composed of planar elements.

Pu and Vosselman [105] combine ground based laser scans and images to extract building facade models that consist primarily of planar arrangments. They use the scan to resolve the general structure of each facade and the imagery for edge detection and texturing. Note although the results are sparse - their system relies on user-intervention (i.e. it is semi-automated).

Barazzetti, Remondino and Scaioni [6] apply multi-view stereo reconstruction techniques to the recovery of dense surfaces representing architectural and heritage objects. Their approach follows the typical structure-from-motion work-flow (esti-

mating orientation parameters, progressive feature matching, least-squares minimisation and bundle-adjustment). The results are qualitatively strong but (though accurate for photogrammetry) the precision of the results is typically inferior to those derived from actively sensed 3D scans.

Wang and Shan [145] tend to the problem of segmentation of unstructured airborne scans for building extraction - in order to combat the loss-of-information induced by discretisation. Positively the method adapts to curved roof-patches however its preservation of multi-scale roof details is lacking.

Golovinskiy, Kim and Funkhouser [46] discuss the design of a system for recognising arbitrary objects in unstructured point-clouds of urban scenes - via a four step process that roughly involves (localisation, segmentation, characterisation and classification). Their approach achieves recall and precision rates of around 65% and 58% respectively.

Calberg, Andrews, Gao and Zakhor [12] present general methods for surface reconstruction from ground-based and airborne laser-scans. This includes a surface mesh merging approach for airborne and ground that exploits the locality of the ground mesh. Positively the surface to surface registration works well and their data-driven approach is versatile enough to adapt to different scanning hardware and variable density inputs - however the dense surface mesh that are generated contain many redundancies (such as planar elements being represented as a multitude of triangles) and possess visible holes at ground level (that result from occlusions).

Kada and McKinley [58] present a flexible and generalised 2D partitioning algorithm to split building footprints into roof-shapes to support construction of LOD2 models from airborne scans. Conceptually this is a strong approach in that it makes few assumptions about the nature of roof-shapes and the iterative decomposition can represent irregular form using simple (typically quadrilateral) shapes. However they do not derive the building footprints (that are decomposed) directly from the LiDAR - instead they rely on pre-existing ground-plans. Nonetheless this a favourable method primarily for the quality of the building models it generates (clean and compact) and the generality of the partitioning approach (as it adapts to simple, concave and complex polygons). Note: this is another method that makes use of parametric primitives to represent 3D masses - however it only includes linear/planar parametrics.

Furukawa, Curless, Seitz and Szeliski [42] propose a fully automated multi-view stereo driven approach to creating navigable building interiors from sets of images. They employ a SfM algorithm designed specifically for Manhattan-world environments in order to recover depth information and exploit the same assumption in the construction of 3D models. Positively their approach enables image-based walkthrough of indoor environments - however the axis-aligned nature of the underlying 3D representations it yields limits its overall utility (in terms of it ability to generalise to non-orthogonal scenes).

Chauve, Labatut and Pons [14] address generalised polygonal modelling from unstructured point-clouds (recovered from passive stereo) via an adaptive division of 3D space using planes (i.e. a polyhedral cell complex). The method can produce mesh that are self-intersection free at variable (user-defined) scales. However it makes restrictive assumptions about the nature of architectural scenes (most no-

tably orthogonal priors) - and can deviate from the input as a result of its hole-filling strategy.

Zhou and Neumann [165] return with a robust generalised approach to top-down building modeling that extends the classic dual contouring method (for crease-preserving iso-surface extraction). 2.5D Dual Contouring is designed to handle arbitrarily shaped roofs by employing a generalised QEF based edge detector on an adaptive grid. Their approach also supports topology-safe simplification but relies on snapping to principal directions to improve surface quality. This is a strong approach in terms of accuracy and robustness (and very much a pioneering method for defining the 2.5D nature of airborne masses clearly) - however little consideration is given to the increased computational expense of evaluating the (volumetric) Hermite data relative to their earlier operators. Nonetheless the algorithm is generalised, purely data-driven and capable of producing high-quality massing without intervention.

Kabolizade, Ebadi and Ahmadi [57] tend to building footprint extraction from airborne imagery using a GVF snake model. Although they demonstrate improved performance relative to previous image snake-models they note that their method does not adapt to multiple building blocks (i.e. it cannot handle roof-shapes only extremal boundary extraction).

Ioannou's thesis [53] deals with low-level computer vision methods applicable to processing unstructured ground-based laser-scanned point-clouds. In particular he proposes the scale-space interest point operator - the Difference-of-Normals and employs local potential well space embedding to address object recognition. Though the DoN is strong from a conceptual perspective, practically the selection of suitable small and large support radii is an ill-defined problem for an arbitrary scan with variable density regions. Note: the LPWSE algorithm employed relies on RANSAC for verification.

Vanegas, Aliaga and Benes [143] use a set of calibrated (oblique) aerial images to automatically construct sparse top-down building masses based on the assumption of three mutually orthogonal directions (x,y,z - i.e. rectilinear axis-aligned blocks). Positively their work employs self-rewriting grammars to good effect - relying one a single generalised re-writing rule (that iteratively refines a cuboid in a coarse to fine manner). However the Manhattan-World assumption limits the scope of buildings that it can correctly represent.

Lafarge, Descombes, Zerubia and Deseillingny [68] extend their earlier DSM reconstruction operators with the notion of a structural approach based on a Gibbs model to control fitting and block assembly. However they note the requirement for improvements to the precision of the parametric primitives that are fitted and the need to reduce computation time.

Sampath and Shan [119] deal specifically with building roof-shape segmentation and polygonisation from airborne scans. Note: the main limitation of this work is that it does not support curvature at the level of building footprints or roof-patches.

Teboul, Simon, Koutsourakis and Paragior [139] employ shape-grammars, supervised classifiers and random walks to extract facade segment descriptors from ortho-rectified images. However they rely heavily on restrictive cartesian priors.

Haala and Kada [49] review automatic building reconstruction methods pre-2010 - with a focus on the approaches applied to create polyhedral building representa-

tions from laser-scanned point clouds as well as from sets of images.

Tang, Huber, Akinci, Lipman and Lytle [136] provide a survey of the state of the art in automatic reconstruction of as-built building information models (BIM) from laser-scanned point clouds. Their review covers methods from computer science and civil engineering, considering the problem in terms of three main types of task: geometric modeling, object recognition and object relationship modeling. They conclude that as-built BIM creation is still a heavily manual process - and document areas where they believe research should be focussed. In particular: the automatic modeling of more complex structures than simple planes and the development of methods that are easily extensible (to new environments).

2010 - 2015

Rusu and Cousins outline the open source Point Cloud Library (PCL)[115] - a collection of point processing algorithms written in C++ with support for multi-core parallelization (using OpenMP or Intel TBB). PCL represents a cross-platform utility toolkit for tasks such as feature-detection, surface-reconstruction, model-fitting and segmentation that is extremely modular (i.e. individual components can be used in isolation) and that integrates into ROS (the Robot Operating System). This development signalled the growing exploitation of 3D scan data for computer perception in the robotics community - and though not directly designed for architectural reconstruction lends itself well to tasks such as door and wall detection (via constrained planar segmentation).

Cheng, Gong, Li and Liu [17] address the problem of multi-modal top-down building modelling from airborne scan and image data. The main limitation of the work is the reliance on orthogonal massing blocks (i.e. the assumption of two perpendicular principal directions).

Zhou and Neumann [166] discuss extensions to the 2.5D paradigm that actively control the topology of polyhedral masses generated by the strategy by considering the associations between roof, wall and point features - enabling topologically safe simplification. Note: their formulation is robust but lacks generality in terms of its suitability to curved components.

Yu, Xu, Liu et al. [158] address segmentation of unstructured urban point clouds via a three stage process of isolating ground points, super-pixelising the remaining salient points and manifold embedded mode seeking.

Lafarge and Mallet [71][72] team up to automatically reconstruct large urban environments from unstructured airborne point-clouds. Their approach (unlike Lafarge's earlier work) does not make use of photographs and as such the accuracy improves - however relative to his earlier parametric and structural approaches model quality suffers (in particular walls are subject to more significant perturbations). The main contribution of their work is integrating automatic terrain meshing and tree approximation methods into the urban reconstruction pipeline so as to yield more complete descriptors of an environment. This is a strong method in terms of its ability to generalise to arbitrary building structures however the non-linear energy minimisation process is quite expensive relative to alternative generalised strategies (in particular they quote processing times in the range of hours for millions of points - which is quite high relative to Zhou's methods for example).

Mathias, Martinovic, Weissenberg and Van Gool [82] document an approach to

inverse procedural modelling of buildings based on determining the correct parameters for pre-defined template grammars. This work can be seen as a complement to structure-from-motion image analysis - in that the input points (used by the detectors) are derived from multi-view stereo reconstruction. The positive aspect of the work is that it yields CGA shape scripts - however the representational scope of the template grammars used is limited.

Kulkarni, Nagesh and Wu [63] address window detection in frontal ground-view images of building facades using projection profiles, mutual information and the snake algorithm (to extract features). The main limitation of the work is that it largely applies to regular facades and is sensitive to imaging artefacts (such as glare) that result from the reflective nature of a window's glass pane.

Martinez, Soria-Medina, Arias and Buffara-Antunes [77] tend to automatic segmentation and feature detection in unstructured ground laser-scans of building facades. Their main idea is to exploit knowledge from the scanner's inclination sensors to orient the point cloud and reduce it to a profile distribution function. This enables facade contours to be extracted (by considering peaks and valleys/troughs) - which in turn facilitates layer based segmentation and planar feature extraction. This is a strong piece of work in terms of the accuracy and level of generalisation of the segmentation - however they note that the precision of the contours is heavily dependent on the resolution (density) of the points.

Pylvanainen, Berclaz, Korah et al. [106] address the challenging problem of textured city model reconstruction from ground based laser-scans and panoramic images specifically for use in augmented reality applications on mobile devices. They employ IMU and GPS readings to register the datasets into a globally consistent coordinate system, alongside structure from motion and skyline detection to tackle the limited range of the LiDAR. This method produces sparse textured facade models with a high degree of visual quality in a matter of days. Though their approach enables accurate street level navigation - (due to the nature of the input data) building roof structures are either flat-tops or omitted.

Zhou and Neumann [167] propose another powerful extension to the 2.5D building reconstruction paradigm in the form of automatically detecting and enforcing global regularities to improve roof model quality for masses recovered from airborne range scans. Their approach dramatically improves the regularisation of models and corrects local fitting errors that may result from noise in a scan. However it is designed to align planar elements only. Curved and irregular roof structures cannot be treated with this strategy.

Alkan and Karsidag [4] analyse the accuracy of TLS sensors by scanning a known set of objects, physically measuring them with callipers and comparing the dimensions represented by the scans to the physical measures. They conclude that the measuring differences are inversely proportional to scanning intensity and directly proportional to scanning distance. They also note the impact of auxiliary features of a surface (reflectivity, colour) on scan accuracy.

Zhou's doctoral thesis [162] provides extended exposition of the 2.5D Dual Contouring strategy including building topology control and automatic discovery of global regularities. This includes the efficient streaming framework for processing large-scale airborne scan datasets.

Ioannou, Taati, Harrap and Greenspan [52] discuss the scale-space DoN opera-

tor, documenting more clearly its applicability to scale-salient cluster extraction in unstructured urban point clouds. Positively their work can be used within pipelines for object recognition and the precision of the clusters isolated (relative to human labelled ground truth data) is shown to be reasonably high.

Riemenschneider, Krispel, Thaller et al. [110] address facade structure parsing from ortho-rectified images using irregular lattices to combat the restrictive nature of orthogonal split-grammars - so as to represent complex arrangements whilst supporting symmetry and repetition.

Dai, Prassad, Schmitt and Van Gool [24] aim to improve the quality of image based facade segmentation operators by learning architectural principles that improve precision at the pixel and structural level. Their work deals mostly with the semantic labelling problem rather than geometric modelling.

Martinovic, Mathias, Weissenberg and Van Gool [79] propose a three-layered facade parsing strategy to recover semantized segmentations from street level images. The key aspects of the approach are a bottom-up RNN to produce semantic segments which are augmented with object detectors (Markov Random Fields defined over the image) and then refactored to enforce architectural principles (such as symmetry and co-occurrence of elements). The method produces reasonable results from imagery however they note that the procedural split grammars they derive from their semantic decomposition are non-parametric (i.e. building instance specific rather than generalised).

Miljanovic, Eiter and Egly [89] tend to the problem of detecting windows in street-level images of building facades. Positively the method is robust to different shaped windows (beyond rectangular) however it will only yield a bounding-box for each window meaning further processing is required to accurately reflect the apertures of irregular windows.

Wu, Agarwal, Curless and Seitz [151] employ swept surfaces to reconstruct clean architectural geometry from unstructured point clouds resulting from multi-view stereo (structure from motion). Their approach accumulates transport and profile curves into floor plans that are converted to surface polygons via the 3D sweep operation. Positively the approach is able to deal well with occluded elements and holes in a cloud and supports detailed surface modelling using displacement maps. However their use of a regular voxel grid (for 3D binning - spatial query acceleration) limits the processing of regions with a sparse number of points. Note: this is a common problem in SfM point-clouds as regions without texture information and at oblique angles to the camera yield fewer points.

Lafarge and Mallet [73] provide a fuller account of their generalised strategy to modelling large-scale city environments from airborne scans - including the hybrid representation employed to handle the versatility present in real scenes.

Martinovic and Van Gool [80] address facade grammar recovery from sets of images using a Bayesian model merging technique extended to the case of 2D languages. Essentially their technique automatically learns attributed stochastic context-free grammars from labelled building facade images.

Dai, Riemenschneider, Schmitt and Van Gool [25] employ a genetic algorithm to synthesise images of facades that bear a high resemblance to an input (seed) facade image (i.e. in a by-example fashion). Like many image based facade operators they rely on a tile based representation (an irregular rectangular lattice).

Van Kerveld, Van Lankveld and Veltkamp [142] produce watertight scenes from urban laser scans and pre-extracted planar surfaces. They use visibility based volumetric analysis through a constrained DT (with graph-cuts) to fill in holes (where there are no polygons) with free-form mesh. Although it can guarantee a watertight return - it cannot represent curvature and the planar detection they employ relies on random sampling. The quality of the resulting models is also inferior to many earlier methods that can also guarantee a watertight return (such as Zhou et al. and Lafarge et al.). In particular walls are poorly represented with visible perturbations.

Lafarge, Keriven, Bredif and Vu [69][70] propose a multi-view stereo reconstruction algorithm for modelling urban scenes from sets of ground-level photographs. The method combines primitives (planes, spheres, cylinders...) with surface mesh to represent regular and irregular features using a multi-label Markov Random Field model. It employs a jump diffusion process to sample primitives and measures quality with a heuristic energy model. Note: this photogrammetric approach relies on iterative refinement to incrementally improve the accuracy of the generated scene representation.

Sun and Salvaggio [129] address top-down building modelling from airborne point-clouds using a hierarchical (divide and conquer) Euclidean clustering strategy. Like pre-existing works they employ region-growing (for segmentation) based on a smoothness constraint and employ Zhou's 2.5D dual contouring method to generate mesh. The results are reasonably effective but oddly only represents flat-roofs.

Hao, Wang, Ning et al. [50] tend to building segmentation from unstructured ground-based laser-scans by mining planes to describe regions with homogenous clusters of points. Their three step process involves segmentation, plane recognition based on properties of the Gaussian image and building identification. Note: although they describe their work as a complete building extraction strategy - it does not actually model surface geometry - it only segments planes.

Truong-Hong, Laefer, Hinks and Carr [141] deal with the challenging problem of extracting accurate facade geometry from unstructured laser-scanned point clouds specifically for computational analysis (i.e. FEM analysis - rather than visualisation as is more common). Their strategy combines an angle criterion with voxelisation - using a kNN search algorithm. Positively the accuracy of the facades recovered by their approach is high relative to manually created CAD drawings and photogrammetric strategies - however they rely on a pre-determined number of kNN points - and the approach does not preserve the true polygonal boundaries of non-rectilinear windows - using bounding boxes instead a proxies.

Raumonen, Casella, Disney et al. [109] propose a tree-model reconstruction operator for unstructured ground-level TLS data - that produces accurate results by representing the global structure of each tree. Positively the results also have high aesthetic quality - however it typically takes a few minutes to process a tree which limits its scalability to large scale urban reconstruction problems.

Weissenberg, Riemenschneider, Prasad and Van Gool [149] investigate the recovery of facade grammars from sets of images so as to construct tailored descriptors that are more concise - rather than trying use a single (one-solution-fits-all) grammar. There approach mixes a binary-split procedure (to create initial parse-trees) with a minimal-rule-length driven virtual synthesis step - and ranks the resulting

procedural rule sets in a bag-of-words fashion. The advantage of their succinct tailored grammars are improved compression and accuracy in retrieval tasks - however they rely upon pre-labelled image data.

Susaki [131] documents a multi-modal approach to producing top-down building models from airborne images and LiDAR data. Positively the method handles segmentation effectively in the presence of building point-clusters that are not disjoint. The key limitation of the work is the restricted nature of the buildings descriptors generated - it only deals with planarity (and in particular relies on a prior-assumption of only four types of quadrilateral component: gable, hipped, flat and slanted roofs) - i.e. it lacks the ability to generalise to complex structures.

Lin, Gao, Zhou, Lu et al. [75] propose a complete system for semantic decomposition and reconstruction of buildings in residential scenes from unstructured LiDAR data. This method produces high-quality results that incorporate non-parallax features (i.e. it yields full 3D models as apposed to top-down 2.5D models) by extracting planes based on the RANSAC paradigm and enforcing structural domain-specific priors (most notably block-wise symmetry and convexity). The powerful thing about this approach is that it is significantly more efficient than pre-existing strategies for unstructured building reconstruction (i.e. piecewise planar surface-reconstruction) - and yields models with strong aesthetic qualities. However their technique is not perfect. Most notably - it is constrained to linear components (no curvature), is designed primarily for residential scenes (optimised for low-rise buildings), does not represent windows and can yield gross inaccuracies due to hole filling (in particular they demonstrate that enforcing block-wise symmetry can have detrimental effects by miss-representing partial data). There is also the issue of the random-sampling taking proportionally half of the pre-processing time of their pipeline. Nonetheless (despite these shortcomings) this represents one of the stronger more recent approaches in that it yields higher quality building assets than previous techniques (for unstructured scans), and embeds rich semantic information.

Musialski, Wonka, Aliaga et at. [95] provide an updated survey and review of LiDAR-based and photogrammetric urban reconstruction methods - including airborne roof modeling operators and ground facade modeling operators.

Wu, Yan, Dong, Zhang and Wonka [152] employ a dynamic programming framework to minimise the description length of facade layout grammars in inverse procedural modelling from images. Essentially their cost function derives more meaningful split grammars than pre-existing solutions based on quadratic programming with linear constraints so as to improve overall alignment and regularise terminals.

Sajadian and Arefi [116] propose a data driven approach to recovering top-down building models from airborne scan data. Their method considers height values, normals, the number of returned pulses, triangle lengths and areas to segment the data - alongside a grid-erosion strategy and RANSAC to detect and extract lines. Due to the manner in which they regularise the results this method is limited to modelling linear orthogonal arrangements - which constrains it generality.

Siddiqui, Teng, Lu and Awrangjeb [125] address building segmentation in aerial images so as to improve the detection of low-lying building features (near the ground terrain, or below a minimum height threshold) in photogrammetric methods. They rely on comparing height differences between neighbouring planes derived from

airborne scans - and use the connectivity information to correct building boundary estimation errors near the terrain.

Tooke's doctoral thesis [140] presents an approach to estimating building energy usage across large areas by exploiting airborne LiDAR to augment conventional simulation driven building energy models. Note: in the strictest sense his research does not deal with building model reconstruction - however it demonstrates some of the wider analytic and policy driving opportunities that result from the use of LiDAR to understand the built environment (beyond the typical as-built model recovery use-cases in construction and asset management).

Xiong, Elberink and Vosselman [154] propose a multi-view stereo approach to recovering compact top-down building models from noisy photogrammetric point clouds (i.e SfM derived). The key aspect of their fully automatic (and parameter-free) approach is favourable extraction of roof-shape graphs (relative to LiDAR based methods) by considering structure points and boundaries. Positively this is a data-driven approach for which the topology of the resulting models is clean (they are regularised) - however this is another plane only method (less generalised than LiDAR based methods) - and its response on curved components of a building's footprint is not addressed.

Diaz-Vilarino, Martinez-Sanchez Laguela et al. [27] tend to the automatic detection of doors in building interiors using photographic images and laser-scan data. Note: they use the point-cloud for detection of each room's envelope upon which they superimpose the images to estimate the location of doors. The point-clouds are also used to confirm and disaffirm the validity of each door candidate (i.e. distinguish doors from similarly sized objects such as book-shelves and other furniture). Positively their approach can detect both open and closed doors - however they note the further requirement to also detect windows.

Mongus, Lukac and Zalik [90] tackle the problem of automatically isolating (segmenting/classifying) terrain and buildings points from airborne laser scans. Their key idea is a multi-scale decomposition by forming a top-hat, scale space using DMPs (differential morphological profiles) on point residuals (which result from approximating a surface using an algorithm they term LoFS - local fitting surfaces - in order to extract planar points). The method yields results comparable to multi-modal detection techniques (image + LiDAR) but only requires points as input.

Fan, Yao and Fu [35] present an approach to building roof-shape segmentation from airborne point-clouds by a hierarchical ridge-based decomposition. However their work is predicated on the invalid assumption of *the fact that every roof can be composed of a set of gabled roofs and single facets which are separated by the gabled roofs*. This massively limits the generality (and hence utility) of the approach. They also rely on RANSAC to mine roof-ridges.

Lafarge's habilitation thesis [65] collates many of his contributions to the field of urban reconstruction. He discusses the various acquisition and reconstructive methods for building modeling from MVS and LiDAR data, the problems that pervade the field and concludes with insights into the future direction of the domain. In particular he cites evaluative tools and measures as largely under developed (due to the challenges in creating ground-truth data, the requirement to share non-public datasets and the lack of effective quantitative criteria that combine both geometric and semantic measures) - and as such as a missing link in terms of bench-marking

operators within the field. He also discusses the manner in which the specialised nature of urban modelling algorithms has tended to result in systems that are highly sensitive to the the input data and less commutable across sensing mediums (i.e. MVS methods do not adapt to LiDAR scans and vice versa). Alongside these recurrent challenges - he points to a key future avenue for investigation - *the creation of a new generic geometric language for modelling urban environments* - as a means to provide a common geometric vocabulary suitable for different data sources and to support inference of 3D models in inverse procedural modelling.

Verdie, Lafarge and Alliez [156] tackle the problem of generating LOD (level of detail) models from 3D surfaces derived from multi-view stereo reconstruction - with a three-step process that involves: classification (into four semantic types of object: ground, tree, facade and roof), abstraction (to regularise planar structures) and reconstruction (to construct mesh at various grains of detail, ranging from LOD0-LOD3). Their classification step relies on horizontality, elevation and planarity to generate a set of super-facets which are labelled using a Markov Random Field. Given the nature of the input data (MVS point-cloud) there results are strong - however they note that LiDAR is more suited to accurate recovery of roof superstructures and facade elements than photogrammetric data.

Abdullah, Bajwa, Gilani et al. [1] propose a method of modelling architecture in unstructured scans of heritage based on dimension reduction (3D to 2D projection), b-spline profile estimation (in 2D) and model construction with surfaces of revolution. Positively their approach employs a more versatile approach to primitive extraction (i.e. n-gonal prisms and data-driven revolutions - over simple spheres, cylinders, cones...) - however it is stochastic in nature (i.e. it randomly-samples) and relies on a user to supply the ground plane to simplify processing. Its applicability to large urban scenes is also unverified.

Yan, Zhang and Zhang [155] construct top-down building models from airborne range scans by extending the conventional 2D snake algorithm to deal specifically with building boundaries by minimising a set of derived energy functions (so as to adjust/regularise each building's representation). The main contribution is a graph reduction technique to simplify each snake to isolated vertices whilst retaining the minimal graph energy. The main limitation is that it only applies to planar roof components.

Choi, Zhou and Koltun [18] present an approach to high-fidelity indoor scene reconstruction using noisy RGB-D video streams from commodity sensing equipment. Their approach relies on improving the registration process for sequential scans by actively suppressing erroneous geometric alignments - even when they outnumber the correct alignments - using robust line processes. In this manner they produce smooth, clean, continuous surfaces whose accuracy surpasses that of pre-existing RGB-D methods. The only limitation of the approach is the density of the surfaces it generates. They are too heavy-weight to represent the interiors of buildings at city (or even site) scale for example. Additionally the surfaces do not embed any semantic associations (i.e. triangle-soup) - and they note that their pipeline relies heavily on global loop closures (in a sensor trajectory/path) to indicate global relations - and the accumulation of drift can distort the reconstruction process. Nonetheless - this is a strong contribution in terms of its accuracy, level of generalisation and registration performance.

2015 - present

Mura, Mattausch, Villanueva et al. [94] propose a robust approach to reconstructing the structure of architectural interior environments from unstructured range scans. Their method relies on extracting planar patches (wall candidates) by considering occlusion - in order to isolate individual rooms (whilst mitigating clutter and missing data. Their main contribution is the application of a diffusion process (inspired by heat propagation) to combat artefacts and imperfections in the input scans. The method yields good results in terms of accuracy and compactness - however they note that it cannot handle slanted walls (i.e. all walls must be vertical), variable ceiling heights (for example duplexes), and it demands all rooms be fully covered by the input scans to ensure completeness. In spite of this, the approach is fast and the planimetric iterative cellular clustering decomposition is a rather imaginative means to address data-driven regularisation.

Cohen, Schwing and Pollefeys [21] address the problem of parsing and semantically labelling images of facades - using a sequential optimisation technique. Unlike prior works that treat the problem as a classification task or that of grammar parsing - they use dynamic programming to boost efficiency by an order of magnitude. The only limitation with this work is its heavy reliance on regularity in a facade. In particular (in the tests they document) the error for low-rise semi-regular residential facades increase significantly relative to the regularised high-rise inner city facades (from $\approx$4%-7% to $\approx$10%-15%).

Martinovic, Knopp, Riemenschneider and Van Gool [78] deal with automatic facade segmentation from SfM point clouds - specifically without dependence on 2D (ortho-rectified) image labelling techniques. Their method operates in 3D *all the way*. The key benefit is the gain in computational efficiency that results (relative to 2D projective schemes) however their 3D facade labelling component (though yielding effective results) applies largely to regularly arranged facades and uses bounding boxes to represent all windows and doors.

Diaz-Vilarino, Khoshelham, Martinez-Sanchez and Arias [28] propose a pipeline for door detection and room-envelope recovery from images and point-clouds of building interiors. The strength of this method is that it generates ortho-aligned images (via ray-based visibility analysis) which are used to texture the walls. However it is subject to more false negatives and positives in door detection than methods based solely on points.

Gimenez, Robert, Suard and Zreik [44] tend to the problem of generating an industry compliant 3D model by scanning and analysing 2D floor-plans. The main limitation of this work (in this context) is its reliance on a pre-existing floor-plan. It does not directly serve the task of building reconstruction even though its input is image data.

Ochmann, Vock, Wessel and Klein [97] address automatic reconstruction of parametric building models from indoor point clouds - so as to yield component based interior room geometry (rather than merely surface-representations which are more common for indoor scenes). They automatically segment the point cloud into rooms in order to remove the outside areas and filter outliers. This is followed by a global optimisation process that resolves wall-to-wall relationships and regularises the results. Essentially they cast the task as a labelling problem - and solve it via energy minimisation. The main strength of this method is its ability to mine

clean compact interior models with door and window representations. However it is worth noting that it involves random sampling processes (such as the stochastic ray-casting used to compute mutual visibility). Nonetheless their results are both quantitively and qualitatively strong.

Daftry, Hoppe and Bischof [23] present an interactive multi-view stereo reconstruction system for creating accurate 3D facade surface models from images captured by micro-aerial-vehicles (MAVs) - by providing real-time feedback that instructs a surveyor as to the quality of the data as it is acquired. They also present a multi-scale camera network method (which limits drift that can result from the incremental model construction) to further increase SfM accuracy. In terms of geometric accuracy - this is a strong photogrammetric method - however in terms of model quality - the dense surfaces that are generated are too verbose (for city-scale reconstruction) and lack semantic information (triangle-soup). The interactive component also has negative implications from the perspective of scalability.

Shahzad and Zhu [122] propose a robust approach to extracting building facade locations for large urban areas using space-borne TomoSAR point-clouds. Note: although this work refers to facade reconstruction - in reality it tends to the problem of wall reconstruction - i.e. detection and polygonisation of vertical features. It does not (for example) actually interrogate or represent the structure of a facade - only the location of vertical walls. Nonetheless this is quite a progressive development in that the ability to robustly extract wall vectors from space-borne scan data enables much wider reconstructive coverage.

Patraucean, Armeni, Nahangi et al. [101] review the state of automatic as-built building information model recovery with a focus on the geometric tasks that underlie such systems (data acquisition and data modelling). They cover computer vision, geometry processing and civil engineering works that play a role in the process, comparing the strengths and limitations of each. They note that the key requirements in the field are new methods for object recognition and the consolidation and integration of existing techniques.

Sun's thesis [130] proposes an innovative solution to large scale building reconstruction from space-borne SAR tomographic scans (at roughly 1ppm resolution). Although he employs some common pre-existing methods (nDSMs, watershed segmentation, hough-line extraction) - the main strength of the contribution is its mitigation of sensing artefacts that present in representations derived from satellites. This in particular opens the door to much wider scale automatic top-down building modelling. However it is worth bearing in mind that the level of detail attainable is reduced relative to methods that employ airborne scans.

Edum-Fotwe, Shepherd and Brown [123] propose a 2D vector shape detection strategy for automatic building modelling from low resolution airborne range scans. The method employs the Hough-transform in concert with an *eat-away* step to vectorise L, T and S shaped building footprints. Whilst the method yields accurate and sparse results - it is specialised for a particular class of building (those that exhibit orthogonality) - and as such lacks the ability to generalise.

Edum-Fotwe, Shepherd and Brown [34] introduce an automatic facade segmentation and reconstruction method for unstructured ground-based laser-scans - that seeks to marry the benefits of data-driven approaches (efficiency and accuracy) with the benefits of model-driven techniques (higher model-quality and semantized

components). Though their method can model the boundaries of irregular (non-rectilinear) apertures, they note that its window and door sash modeling requires refinement.

Zheng, Weng and Zheng [161] address top-down building model reconstruction from medium resolution airborne LiDAR - by employing a hybrid strategy that aims to be efficient and robust at low resolutions. Their key contribution is a novel ridge detection method - which deals well with the limited information provided by the lower-res. nDSM ($\approx$ 0.9m point spacing). However relative to earlier *hybrid* schemes, their approach does not support free-form mesh.

McClune's doctoral thesis [86] tends to the problem of automatic 3D reconstruction of buildings models from dense image-matching datasets. His key idea is to focus on extracting corners in order to combat the problems inherent to plane fitting. Whilst he demonstrates good corner detection results for top-down architectural images - his approach suffers from limited response in terms of accurate roof-shape projection. Essentially there is not enough information provided by his corner-connectivity techniques to accurately convert the 2D vector-arrangements extracted to 2.5D building masses - and hence the precision of the results (relative to LiDAR based strategies) is lacking. Additionally there are also problems related to assuming ubiquitous linearity (i.e. poor generalisation to buildings exhibiting curvature).

Bacharidis, Sarri, Paravolidakis et al. [5] address 3D facade model reconstruction from image data - through a multi-modal fusing of stereoscopic and tachometry data. Their work aims to deliver more realistic reconstructions than prior photogrammetric approaches. The approach relies on extracting a 2D skeleton of a building by active contouring and Hough line extraction. They then infer the structural details of facades using a depth derived stereoscopic layout. Note: their work relies on merging the structural data derived from the images with geo-referenced points. They also employ co-linearity and parallelism conditions to improve line extraction. Positively this method can produce sparser facade geometry - however there is little semantic information embedded in the representations it generates (in particular there is no differentiation between walls/windows/doors or types of objects upon facades - the result is a surface rather than component-based).

***

In closing: the domain of architectural reconstruction has come along way over the last three decades - and whilst there remain certain recurrent problems - as a collective we are gradually closing the gap between the capabilities of human CAD technicians (who create architectural models manually) and automated modeling algorithms (that reconstruct sampled data). Whilst many technical challenges still lay ahead - the future looks bright for parties who rely upon accurate geometric representations of the built-environment. As scanning hardware and sensor technologies advance the scope of that which can be accurately measured has grown from simple block masses to highly detailed architectural representations - capturing not only roof-geometry - but increasingly street-level facade and building interior geometry. However in order to keep-up with such development (and indeed to exploit laser-scanning to its fullest) the software patterns employed to construct meaningful geometric models from architectural point-clouds must also advance.

Next we'll focus on a subset of these methods - delving deeper into their oper-

ation, considering the domain-specific observations and insights they exploit and the practical trade-offs that pervade the use of each.

1.1.3 Key Related Work: Case Studies

This portion of the literature review, details key case-study reconstruction methods. The algorithms discussed represent archetypal and highly relevant pre-existing approaches to addressing reconstructive problems related to architectural modelling. Some of the algorithms operate on point clouds, others on sets of photographs. For each case-study, a brief synopsis is provided. This is supplemented by an outline of the contribution and the key insights drawn from the research.

The strengths and limitations of each method are also outlined in each of the succeeding discussions. Note: these are the key reconstructive approaches that inspired the developments presented in the second part of this thesis.

CS1: Zhou and Neumann - 2.5D Dual Contouring + Extensions
Data-Driven and Analytic : [167], [164], [165], [163], [166]

The first case study reviews one of the most influential airborne building reconstruction methodologies to have been proposed in recent years.

Zhou observed that one of the key distinguishing features of manually constructed models - relative to reconstructed surfaces - is the presence of sharp (crisp) edges - and that the pre-existing Dual-Contouring algorithm of (Tau et al. [56]) could be refactored in order to preserve such features in top-down point-clouds. Zhou also proposed several key efficiency optimisations - such as the streaming strategy for large-scale reconstruction - that ensured that the resultant building reconstruction algorithm could be executed on commodity hardware within a reasonable amount of time. This (along with its extensions) represents the dominant analytic strategy for automatic top-down building model reconstruction from laser-scans.

CS2: Lafarge, Mallet et al. - Hybrid Airborne Scene Reconstruction
Data-Driven + Model-Driven and Stochastic : [71], [72], [73], [156], [67], [68]

The second case study reviews a generalised strategy for the construction large urban environments from airborne laser-scanned and MVS point-clouds.

Lafarge et al. determined that a conglomerate reconstruction operator was one of the only stable means to mitigate unsatisfactory results for highly irregular architectural models. They realised a hybrid approach to building reconstruction was required to deal with the diversity present in the real-world. They observed that a mix of stochastic sampling, analytic solving and template fitting provided a robust balance - especially in the presence of low-resolution input. In terms of top-down building reconstruction from airborne laser-scans - this body of research represents the dominant stochastic strategy. The key aspect to take away from this is the versatility of a generalised cobglomerate strategy. However it is worth noting the relative expense (from a computational perspective) of the strategy relative to more efficient approaches such as that of Zhou and Neumann.

CS3: Calberg et al. - Airborne and Ground Surface Reconstruction
Data-Driven and Analytic : [12]

This case study reviews a multi-modal photogrammetric building reconstruction

method for the construction of dense textured building surface geometry.

Calberg et al. were amongst the first to propose a fully automatic merging strategy that could operate over multi-modal urban data. Now although the geometric quality of the results produced are inferior to more recent multi-modal merging operators - the great strength of their method is its fast execution time. In particular their algorithm is designed to scale to large datasets effectively. Additionally unlike more recent techniques this a purely data-driven method which makes it incredibly versatile from the perspective of creating accurate assets.

CS4: Müller et al. - Image-based Procedural Modeling of Facades

Model-Driven and Analytic : [93]

This case study reviews a highly influential facade construction approach that creates textured sparse 3D models from ortho-aligned images of building facades.

The main insight drawn from this research is that the layer of abstraction injected by a suitable high-level facade representation (which in this case is a rectilinear tree-based split-grammar) facilitates the recovery of higher quality facade models than would otherwise be feasible from a single image. However the caveat to this is the reduction in accuracy relative to data-driven methods that operate on actively sensed point-clouds. The major point is that this archetypal approach (which has sired numerous derivative approaches) yields facade models that look good and are suitable for interactive visualisation tasks, but that can not necessarily be relied upon for analysis of the built environment. In particular representation of irregular facade-layouts (i.e. those that do not obey the rectilinear constraint) and irregular apertures (i.e. non-orthogonal window and door frame boundaries) is missing.

Nonetheless the notion of mining a high-level generative descriptor of a facade (rather than merely a set of edges or bounding boxes) is one that has had a large influence on the ground-scan reconstruction research presented in this thesis.

CS5: Martinez et al. - Automatic Facade Segmentation from Ground Scans

Data-Driven and Analytic : [77]

This case study reviews a specialised data-driven facade segmentation operator for unstructured ground-level laser-scanned point-cloud data.

The main observation is the accuracy and versatility of the facade segmentation method employed. Their method is based on statistical analysis of the depth data rather than priors related to common facade layouts. As such it is able to isolate high fidelity facade layers. The results are not only visually compelling, they are precise in terms window and door detection and localisation. This enables the dimensions of facade features to be extracted from the segments to within a tolerance of a few centimetres. Essentially their iterative profile-extraction and thresholding strategy enables precise contours to be recovered - most importantly not only for windows and doors, but also for balconies, buttresses and adornments.

However the main challenge in adapting this work to larger-scale ground-based scans that contain facades is that it relies on exact knowledge of the scanner's position and inclination properties relative to the facade. It is hard to determine how well this strategy adapts to scans for which such information is not present. Additionally their results consider high-density facade scans for a single facade which (though exhibiting irregularity) calls into question the stability of the method

for lower-density scans (such as from larger distances or SLAM based capture).

Despite this, beyond the conceptual strength of the layered analysis, the approach actually works - and in many regards sets the standard in terms of the precision and level of detail attainable for decomposition of unstructured facade points.

CS5: Lin et al. - Semantic Reconstruction of Residential Scenes
Data-Driven + Model-Driven and Analytic + Stochastic : [75]

This case study reviews an important more recent approach to high-quality reconstruction using ground laser-scans that employs multiple paradigms.

The vital point is that despite the fact the execution times are quite high (at roughly 2 minutes per building - which though an improvement on previous approaches which take 15-20 minutes per-building - is still inadequate for large-scale reconstruction) - the key positive feature of this method is the higher quality models it generates - through its enforcement of structural priors (in particular block-wise symmetry). It can represent non-parallax features where purely top-down strategies can not - however it is not as generalised as the dominant 2.5D approaches in that it is designed primarily for low-rise residential buildings without curvature. Additionally despite the fact ground data is used as input the technique's focus is mass-reconstruction and the level of detail upon each facade is minimal - (i.e. subtle details such as stairways, window-frames, hand-rails are not represented). Nonetheless this is a strong approach from the perspective of improving building model quality. Note: however though that Lin et al. discuss the fact the process of RANSAC (to extract planes) consumes a large portion of the overall runtime.

CS6: Wu et al. - Schematic Reconstruction
Data-Driven and Analytic : [151]

This case study reviews a particularly interesting schematic reconstruction approach for building reconstruction from photogrammetric point-clouds.

This method is particularly noteworthy for its ability to preserve subtle parallax displacements whilst still yielding largely regularised swept profiles. However the main observation and insight drawn from it is the versatility of the 3D sweep (i.e. the generalised cylinder) in the representation of architecture from sampled data - even lower-quality photogrammetric point-clouds. Wu demonstrates that a large portion of architectural surfaces can be approximated by sweeping polygonal profiles about rails - and that further more - the descriptors for each such components can be very concise - (i.e. two polyline paths). The other benefit of accumulating profiles to feed into a GC generator is the inherent mitigation of missing data. However few building reconstruction operators support the extraction of 3D sweeps.

Note: Wu's research has (like all these case-studies) influenced the airborne and ground reconstruction operators documented in this thesis. In particular the top-down mass-model reconstruction operator (in chapter 3) uses data-driven sweeps to represent terraces, whilst the facade reconstruction operator (in chapter 4) exploits them to represent and generate irregular window and door models.

***

This portion of the literature review revised the key pre-existing methods for 3D

building model reconstruction from sampled representations. Whilst the focus of this particular research is reconstruction from laser-scanned point-clouds this section also discussed photogrammetric approaches (i.e. MVS and SfM based) so as to provide a fuller account of the various ways in which previous researchers have tackled the general problem of 3D architectural reconstruction.

1.1.4 Limitations and Recurrent Problems

This part of the literature review revised the prior computational approaches in active-sensing - as they relate to 3D reconstruction of the built-environment.

This closing portion summarises the key limitations and recurrent problems - identified as a product of traversing the pre-existing literature - that relate to the core problem of efficiently reconstructing semantically-rich compact architectural geometry from structured and unstructured laser-scanned point-clouds.

- **Sensitivity to Sensing-Noise** - relative to general surface-reconstruction.
- **Low Tractability of Results** - particularly for obfuscated operators.
- **Preservation of Multi-Scale Features** - i.e. at variable spatial scales - this is a particular problem for plane-based methods.
- **Computational Efficiency** - large execution times for city-scale datasets - specifically non-linear growth in execution time.
- **Un-Intuitive Control Parameters** - which ties in to intractability.
- **Non-Deterministic Geometric-Results** - as a product of random sampling strategies such as RANSAC - which limits repeatability.
- **Limited-Ability-to-Generalise** - to irregular architectural forms.
- **Robustness/Versatility of Priors and Heuristics** - the heavy use of orthogonal and linearity constraints that do not generalise to irregular buildings.

In essence, one could sum up the current state of the domain of architectural reconstruction by saying that: *many low-level problems have been addressed well - whilst several higher-level problems are yet to be definitively resolved*.

1.2 Procedural Modeling of Architecture

Procedural modeling is a vast sub-discipline of procedural content generation - that focusses on the algorithmic definition of geometric models. Procedural modeling spans the application of: self-rewriting shape grammars, fractals and L-systems, constructive modeling, parametric modeling, noise sythesis, by-example modeling and similar generative abstractions that enable the concise definition of organic and man-made geometric structures.

As such it is beyond the scope of this review to cover all aspects of, contributions to, and developments within the domain. Instead the aim is to highlight the key strategies to procedural modeling of architecture that informed this research.

1.2.1 Outline of Contributions and Developments

Early influential work that addresses procedural methods for creating cities includes Parish and Müller's [99] automatic system (that addresses layout and subdivision of lots and road networks), and the introduction of parametric set grammars by Wonka, Wimmer et al. [150] (in particular the definition of split-grammars as a tool for facade representation).

Later on Müller, Wonka, Haegler, Ulmer and Van Gool [92] introduce CGA shape a novel shape grammar for the generative definition and construction of procedural building models. This work extends the concept and applications of split-grammars. Slightly later Kelly and McCabe [59] present an interactive city modeling tool that employs parametric modeling strategies.

Other researchers such as Lin, Cohen-Or Zhang et al. [76] have dealt with the problem of retargeting irregular architectural models (i.e. procedural style-transfer).

More recently Schwarz and Müller [121] propose an advanced variant of CGA-Shape (the dominant architectural shape grammar), whilst Fan and Wonka [36] devise a probabilistic model for modeling the exteriors of residential buildings.

The current research trends in procedural modeling of architecture include sketch-based interactive generation of urban models (such as [96]) and automatic grammar-refactoring (i.e. operations such as rule-length optimisation).

1.2.2 Automatic Building and City Model Generation

In terms of procedurally generating entire city-models, the irrefutable leader in the domain is Pascal Müller and his revolutionary approach to describing complete building exteriors using self-rewriting Chomsky grammars. Prior to his seminal works [99], and [92], the applications of shape grammars where often limited to the production of models to represent organic assets (such as L-Systems for modeling plant growth). However Müller demonstrated that due to the repetition and self-similarity present in architecture one could re-appropriate the self-rewriting nature of a formal shape grammar in order to define and generate coherent high-quality rule-based man-made assets. Further more, one of the key differences between Müller's approach and pre-existing building generation methods was the application of split-grammars for defining structured sub-divisions in facade modelling. This distinctive feature (combined with the underlying Chomsky grammar) enabled highly-varied facade models to be constructed as the product of rectilinear

(cell-based) recursive assignments. Put simply Müller not only devises a suitable (flexible and generalised) building representation, but he also exposed the intrinsics of the representation through a AEC specific programmable shape grammar. In this manner Müller's research laid the foundation for rule-based city-model generation. Müller's lasting contributions to the domain have included:

- Applying Self-Re-Writting Chomsky Grammars to the definition of hierarchical architectural component arrangements. Note: that although other research has employed similar approaches - much of it dealt with the 2D-case of generating a plan algorithmically - whereas Müller's work addressed the much harder case of their extension to 3D.
- Devising Parametric Lot-Division routines which included support for generating Manhattan (gridded), radial and organic city-blocks. Note: the thing that distinguishes City-Engine from the related research is the ability to use user-supplied geo-spatial data to guide the process.
- Applying Planimetric Split-Grammars to modelling structured architectural facades and the associated functions for representing phenomena such as axis-aligned repetition and symmetry.
- General-purpose AEC shape-grammar (CGA-Shape) that acted as a basis for the stand-alone City-Model generation system - City-Engine.

For these reasons, Müller is (quite rightly) held as somewhat of a hero within the domain and his research stands as a shining example of how powerful procedural model generation can be when applied at scale. One of the most powerful aspects of Müller's facade split-grammar is its suitability to backward-chaining generative modelling tasks. In particular he demonstrated that one could also use the same rectilinear split representation to facilitate procedural modelling of facades from sampled representations such as ortho-aligned facade photographs. [93] This trend still continues today, and numerous researchers (including [80], [25], [110] and [152]) have applied the semantics of his facade split-grammar to the problem of inverse-procedural modelling of architectural facade geometry.

Since its inception City-Engine's feature-set has grown to include: profiling and reporting rules, automatic dimensioning, semi-automatic modelling, integration with the Arc-GIS framework, support for a wide array of model export formats (including FBX, OBJ and RIB to name but a few). City-Engine is now considered the de-facto standard procedural city-model generator within AEC, GIS, Energy-Markets and Digital-Media. Most recently [121] an updated version of the underlying shape-grammar (CGA-Shape) added support for boolean logic operations which enables constructive solid mass modelling directly within City-Engine.

Now - although it is an incredibly powerful tool - research and experimentation unveiled a number of minor problems with CGA-Shape and City-Engine's implementation that (despite being non-critical) have the potential to limit its utility.

- As the complexity of architectural arrangements being defined increases the brevity of CGA-shape diminished.
- Reliance on external (manually-modelled) assets to represent features such as windows, door and adornments.

- Limited support for realtime procedural modelling within interactive simulation environments such as game-engines.
- Suitable for modelling architectural exteriors (outside buildings) - but lacks support for modelling architectural interiors (inside buildings).

Despite these minor short-comings Müller's research and subsequent industrial developments remain the most comprehensive solution to producing high quality city-models algorithmically. The important observation is that although Müller was not the first, or the last researcher to address the problem (refer to [59] for an example of a simpler parametric alternative), it is his approach that has stuck and that continues to stand the test of time. This is because he recognised not only the requirement to describe hierarchical arrangements algorithmically but further directly exposed a means for others to add new procedural objects by defining rules. This meant that City-Engine could be programmed to fit a technicians desire rather than being constrained to a finite set of built-in types. I cannot stress enough how powerful this ideology is and indeed how inspiring it has been to this work. Müller's research dealt with a long standing-problem in procedural content generation - that of expandability. City-Engine makes it easy for a technician to define new dynamic generative architectural functions without having to worry about the nitty-gritty of managing computational resources or coordinating boiler-plate geometric and spatial tasks. Whilst City-Engine remains (to date) the most comprehensive solution for procedural modelling of architecture - alternatives exist.

Note: whilst procedural building generation is a research topic that has received substantial attention over the last two decades (and one that plays an important role in this research) it is not (in the strictest sense) the focus of this research. Nonetheless an awareness/appreciation of the forward-chaining generative patterns that pervade the domain is useful from the perspective of understanding the types of modelling abstraction that one might wield to combat backward-chaining (reserve-engineering) problems. The main consideration to bear in mind is that high-compression, generative building modelling techniques are typically not designed to facilitate direct specification of an object's representation in a data-driven manner, which means that (often) the task of identifying a grammar to precisely represent a concrete instance of an object could be considered that of coercing a less suitable implicit representation when a direct explicit representation would be more suitable. Essentially forward chaining patterns enable exploration of design space - but are harder to apply to reverse-engineering problems.

This is one of the key aspects that distinguishes Müller's research from pre-existing (and many recent) approaches - i.e. its un-common applicability to backward chaining tasks. From this one might draw the high-level conclusion that within procedural modelling the strongest abstractions are those that are flexible and generalised enough to serve both generative and reconstructive pursuits.

1.2.3 Limitations and Recurrent Problems

Due to the fact that the domain of procedural modelling draws heavily upon (and as such can be largely rationalised in terms of) the fields of Computational-Geometry, the study of Algorithms and Data-Structures, Language-Theory and Computer-Graphics - the large majority of low-level problems have already been solved or

addressed within each specific domain. What this means is that whilst problems such as maximising the scope for data-amplification, enabling artistic control, ensuring generality, flexibility and usability still play a large role, in practice the prominent problems within procedural modelling relate more to exploring concrete applications of the patterns defined within the domain. Essentially the most pertinent problems identified by this literature review relate not to forward-chaining procedural modelling, but rather to backward-chaining procedural reconstruction.

However in-spite of this general observation there is one common underlying problem that prevails amongst procedural modelling strategies and abstractions that are in-direct or declarative (as opposed to imperative) in nature. This is the ability to exercise fine-grain control of an algorithm - and is sometimes referred to in the context of making local edits. The critical observation is that higher-level abstractions make it harder to specify unique and irregular features than lower-level (direct) techniques - because often manipulation of a high-level grammar can have un-intentional global effects beyond the local revision sought. Although this is well documented in the literature - and many pre-existing researchers have addressed it within particular contexts (i.e. for organic modelling or modelling furniture) - there remains a lack of a generalised solution for the case of arbitrary man-made objects. Ultimately - in procedural modelling there is often a fundamental trade off between the precision of control afforded by an abstraction and the brevity of its representation. The more concise the representation the less artistic control - the greater the scope for explicit control the less succint the representation is.

1.3 Research Summary

Having reviewed the pre-existing literature in architectural reconstruction and procedural modelling, this section summarises the vital pre-existing academic contributions to both fields that relate to the core-aims of this thesis. This section then summarises the key problems within the domains of point-cloud processing and architectural-reconstruction that were identified by traversing the literature.

Key Research in Architectural Reconstruction

In terms of the vital contributions to architectural reconstruction the following revises the research most relevant to achieving the aims of this project.

- 2.5D Dual Contouring, Tile Streaming, Topology-Control, Global Regularities - Zhou and Neumann [167], [164], [165], [163], [166]
- Building Large Urban Environments from Unstructured Point-Data, Hybrid Method - Lafarge, Mallet et al. [71], [72], [73], [156], [67], [68]
- Automatic Processing of Terrestrial Laser Scanning Data of Building Facades - Martinez et al. [77]
- Semantic Decomposition and Reconstruction of Residential Scenes from Unstructured Laser-Scans - Lin et al. [75]
- Image-Based Modelling of Facades - Mueller, Wonka et al. [93]

Key Research in Procedural Modelling of Architecture

In terms of the crucial contribution to procedural modelling of architecture, the most advanced development is the result of research by Müller and Wonka [92], [99], [93], [121] in devising and continuing to develop City-Engine and the CGA-Shape grammar. Note: that whilst procedural modelling of architecture is still a highly-active field of research - it is fair to say that there already exists stable solutions that address it effectively. Beyond City-Engine a myriad of simpler parametric building generators exist that integrate into frameworks and content-authoring solutions such as Unity-3D, Unreal-Engine, Rhinoceros, Modo and Blender (to name but a few). Further in industrial management of the entire life-cycle of buildings - BIM solutions such as AutoDesk's Revit expose a plethora of generative components for constructing 3D building assets procedurally - as well as defining relationships between assets and simulating their physical behaviour.

In short - forward chaining (generative) modelling of architecture can largely be considered a solved problem. However backward chaining (inverse/reverse engineering) modelling of architecture from sampled data still poses many challenges.

Open-Problems in Point-Cloud Processing

This section summarises the open-problems in point-cloud processing.

- Automatic Segmentation: in particular the heavy use of plane-only segmentation methods (many of which rely on random sampling for plane estimation) and the use of interactive processes to combat over and under segmentation.
- Sparse Reconstruction: most notably the lack of generalised structure-preserving simplification and approximation methods for dense surface representations derived from laser-scanned point-clouds of arbitrary objects.
- Computational Efficiency: the growth in execution time as a product of the number of input points. Specifically the typically non-linear computational complexity of point-processing methods. This remains a notable issue especially for systems that operate on unstructured relative to structured scans.

Open-Problems in Architectural Reconstruction

This section summarises the open-problems in architectural reconstruction.

- Improving Automatic Model Quality: this is particularly notable for photogrammetric building reconstruction methods - however LiDAR based methods are also plagued by similar (though typically distinct) model quality issues.
- Balancing Accuracy and Brevity: model-driven techniques typically yield sparse approximations of buildings with a greater amount of error - however more accurate data-driven methods tend to bloat the size of build representations.
- Recovering Semantized Models: i.e. the gap in the manipulation and editing facilities that exist between manually and automatically generated building models. For example manually created models typically allow high-level revision and manipulation - whilst auto-generated models do not and (irrespective of the internal semantics they apply and enforce) yield sets of triangles.
- Reliance on Intermediary Interactive Editing: represents a major problem from the perspective of scalability - in the sense that for as long as a human is required (in-the-loop) to supervise the execution of an algorithm - it is difficult (if not impossible) to scale the scope of a region processed efficiently.
- Algorithmic Scalability: from the perspective of computational efficiency (i.e. irrespective of user interaction) - and in particular the non-linear growth in runtime typically associated with the methods that employ random-sampling. As previously noted this problem is not specific to building reconstruction - however specifically within this context it manifests as systems taking hours or days to process city-scale assets that could be handled by general surface-reconstruction and mesh-approximation methods within minutes.
- Algorithmic Evaluation - Limited Access to Implementations and Bench Marking Datasets: is still a major problem in building reconstruction form laser-scanned point-clouds - despite the last three decades of technical progress. This manifests as fewer standardised testing and profiling datasets relative to photogrammetric techniques - and makes the process of comparing operators harder (if not impossible) within the domain. Few techniques undergo the same level of rigour in evaluation relative to less specialised tasks in computational geometry, computer graphics and computer vision.

Part II

Development

Chapter 2

Semantic Change Detection

What is it?

A multi-modal semantic change-detector for city-scale urban models.

Why does it exist?

To reduce the execution time associated with reconstruction of large-city-models by enabling 'reconstructive-culling' (also referred to as 'selective-reconstruction'), and detect and categorise geometric modelling inaccuracies.

How does it work?

By detecting and classifying mutually-exclusive instances of architectural variance using tractable shape predicates. Each predicate helps to identify extensions, reductions, re-positions, constructions, removals and replacements in out-dated CAD models, using newly-acquired up-to-date laser scans.

beyond point-to-plane...

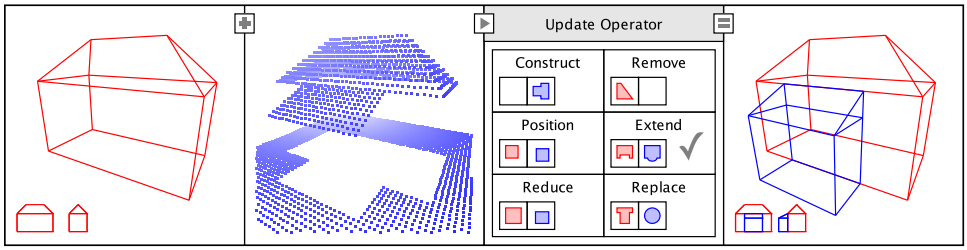

Figure 2.1 : An overview of the role of the semantic change detector - from left to right: (red) an out-dated CAD model, (blue) an up-to-date aerial-laser scan, (grid) determining the class of variance between the two geometries and (red and blue) the result of merging the detected extension with the out-dated mass.

2.1 Overview

The aim of this chapter is to define and analyse an operator capable of providing high-level (actionable) descriptions of geometric variance that could be used to guide a CAD technician in manually updating a city model or an automatic algorithm in architectural modelling from laser-scan data. One may wonder - why begin by addressing an analytic problem that does not inherently yield geometry? Recall that the literature review drew attention to one of the greatest issues present when automatically processing mid to large sized point-sets - namely scalability. The analytic operator defined in this chapter is targeted specifically at addressing this.

It achieves this by enabling *selective-reconstruction* (or *reconstructive-culling*). Selective reconstruction is simply a term the author has dubbed to refer to optimising the runtime of a geometry recovery algorithm by analysing and only re-modelling buildings that exhibit significant temporal variations. In layman's terms the aim is to ensure that a reconstruction algorithm only spends compute time updating buildings that actually require updating (because their geometric form has changed). As an optimisation it relies upon the observation that often only specific buildings or sites in a city need to be altered to bring an out-dated city-scale CAD model up-to date. The optimisation is analogous to view-frustum culling (in object-order rendering algorithms), and in the same manner ensures redundant work isn't undertaken during the task of temporally updating city models.

One could argue that this optimisation bears little relevance to procedural modelling from point-sets and is simply a generic culling strategy. Though this would be true for most domains, in the case of architectural modelling this is not the case. This is because selectively culling the reconstructive process allows an algorithm to combine pre-existing geometry with newly-recovered geometry. Such a process could not be trivially automated without a means to compare and understand the geometric variance between an old dataset and a new dataset. In essence without a change-detector, an algorithm has no choice but to reconstruct everything in the input. The presence of a semantic change-detector allows an algorithm to be selective about what it tries to model and as such trivialises the integration of high-quality manually constructed models with accurate sparse procedural-models.

This is a key feature of this optimisation. Beyond merely the efficiency, it allows an operator to take advantage of pre-existing CAD models. This means it need not waste previous man-made building models that may already have been optimised for rendering. It can exploit the uncountable man-hours of labour previously undertaken (to manually construct a city model) and incorporate newly-recovered geometry only when it is required.

Additionally there is general worth (both to geometers and architects) in the ability to automatically identify physical areas that have changed.

To help clarify examples of concrete potential use cases for such a multi-modal semantic geometric change detector, one could conceive using it:

1. As a pre-processing optimisation and segmentation stage in an automatic modelling algorithm (such as that proposed in this thesis)
2. As an interactive variance visualisation program for CAD technicians, Architects and 3D Modellers, used to guide manual city updates

3. As a revision-agent (a tracker and classifier) in a BIM style city planning or development documentation framework

For reference the key objectives, requirements and behavioural desires of the operator are stated (remember this is the method of understanding the changes between different types of geometry - in this case CAD and LiDAR).

- Efficacy - in distinguishing the classes of deviance that are of interest to a practitioner working with temporal building assets.
- Efficiency - for obvious reasons. This only works as an optimisation if it reduces the time associated with city-scale reconstruction. Although one could argue it still has benefits since it enables pre-existing models to be included (and composited alongside newly recovered geometry), fundamentally the desire for this pre-process is to execute in proportionally less time than the procedural modelling stages.
- Robustness - to the presence of noise in the sampled dataset and geometric degeneracy in the CAD model. Since there are few guarantees about the nature of the input datasets (such as the resolution, distortions, or the presence of non-manifold entities), the classifier should be capable of performing analysis in the presence of such artefacts.
- Intuitive - to understand and implement. Though this does not change the result - it will inadvertently have an impact on the accessibility and overall utilisation of the operator. The easier the operator is to conceptualise and implement, the greater the likelihood it will actually be implemented by other researchers and hence expanded upon.
- Tractability - of the result. Given the potential for subjectivity in the outcome - it is imperative that the class associated with each out-dated building be unambiguously traceable from the implicit and derived properties of the building and its pairwise match in the LiDAR.

The remainder of this chapter is structured as follows:

- The methodology section defines the contribution in terms of the key stages employed to classify actionable geometric variances between old-CAD models and new-point-clouds. It covers the 6 classes of variance recognised (construct, remove, replace, position, extend and reduce), detailing for each the underlying rationale and formalism.
- The experimental results section enumerates the outcomes of profiling the performance and behaviour of the change detector using an out-dated CAD model and manually labelled ground truth data for the city of Bath. It also discusses the operator's performance on synthetic datasets - which clarifies abstractly where the method fails - i.e. the limitations of the approach.
- The analysis and evaluation details high-level analysis of the change-detection operator. The section considers and expands upon the results and presents insights drawn from the experiments.
- The discussions and summary section provides a synopsis of this chapter - reiterating the aims and outcomes and commenting on the implications going forward. It seeks to be a succinct - with bullet-points used to revise the key-points, ideas and concepts introduced.

2.2 Background and Context

This section details the theory pertinent to semantic change detection.

2.2.1 Key Related Work

This subsection discusses the key related work in more detail. The focus is characterising the behavioural properties of the methods that inspired and informed development of the semantic-change-detector. It considers the strengths and limitations and how they relate to the problem addressed in this chapter.

Segmenting Aerial Datasets

Recall that segmentation algorithms can vary greatly based on the nature of the data they operate on. Prominent pre-existing techniques that involve segmenting aerial LiDAR (with a focus on building extraction) includes Sampath et al. [118] Matei et al. [81] and Wang et al. [145] In the case of structured range images such as aerial DEMs and RGB-depth scans, the greatest benefit is the processing efficiency derived from implicit knowledge of each points neighbourhood. Their vital flaw however lies is the loss of information. In the case of unstructured range images (such as radial fixed-position and vehicle-mounted scans) the key advantage is the level of detail that can be recovered for each object - as well as the lack of topological constraints. However the downside of segmenting unstructured point-sets is correspondingly large runtimes. Notable approaches include the work of Dorninger et al. [31], Carlberg et al. [12] and Rabbani et al. [107]. We make special mention of the difference-of-normals - which is a multi-scale operator for analysing and segmenting unstructured scans (the result of research by Ioannou et al. [52]). Inspired by the Difference-of-Gaussians (from image processing) it provides a means of identifying clusters of points at equivalent feature-scales, by considering the variation in the unit normal when each point's normal is computed with two different support-radii for neighbourhood determination. In the field of Computer Vision, a commonly used approach is graph-based segmentation[134]. Graph-cutting (and its variants) form the basis of numerous image segmentation approaches. The work of Felzenszwalb et al. [38] focuses on maximising the efficiency of such methods. Another widely adopted methodology is MSER (Maximally Stable Extremal Regions) heavily used in video segmentation. However when applied to the range data both approaches failed to yield suitable levels of object recall. In particular MSER proved temperamental and frequently omitted smaller building masses. The Difference of Elevation Models (DoEM) is an informal approach to extracting objects in urban range data based on the removal of the ground and clustering of remaining disjoint homogeneous collections of points. Though no single work claims credit for its definition, the principal is evident in the work of Lin et al. [75] (as an component of their method of semantically decomposing residential LiDAR scans) and Carlberg et al. [12] (in their method of segmenting and reconstructing complete surface descriptors from aerial and ground based range data). The key requirement of the DoEM paradigm is a corresponding terrain model for the surface model being segmented. Acquiring terrain models (via approximation or surveying) has a well established history within the photogrammetry and remote sensing communities and the work of Kraus et al. [62] provides modern advanced methods of automatic terrain model generation from digital surface

models. However be aware that - because terrain models are generally approximated from surface models - they are subject to higher error-bounds. Essentially the scope for error is typically larger in a terrain model than an surface model.

Change Detection

Recall that determining regions of conflict between similar (but distinct) geometric data sets pervades a number of modelling and analytic problems. For example, variance detection is a fundamental requirement for higher level problems such as the generation of varying levels of detail within Computer Graphics and calculating the approximation error within Computer Vision - where measures such as the Hausdorff distance (a measure of the maximum point-to-edge/face distance between two geometries) may be used to compare template data to real data. At a high level the key attributes are the topology and modality of the input operands, the distance-metric employed and the grain-of-comparison. To illustrate this - one can consider the work of:

- Memoli et al. [88] as a point-to-point (mono-modal) isometry invariant object-level operator for comparing and analysing the differences between point-clouds representing manifolds.
- Chang et al. [13] as a plane-to-plane (mono-modal) deformable object-level operator for comparing the differences between topologically equivalent meshes in different poses recovered from laser scans. Their approach registers the surface geometry based on volumetric analysis.
- Girardeau-Montaut et al. [45] as a point-to-point (mono-modal) rigid body operator for low-level computationally efficient point variancing - specifically designed to operate on unstructured-ground-laser-scans.
- the proposed Semantic-Change-Detector as a point-to-plane (multi-modal) rigid-body object-level operator for comparing and classifying the differences between polygonal-meshes and discretised displacement-fields.

The positive thing about the problem of differencing is that is has a well-established history (for images, point-sets and polyhedra-mesh), from which to draw inspiration. Jones et al. [55] also provide a comprehensive survey of the dominant methods of processing point-sets as distance-fields. Fundamentally variancing (equivalent to differencing) is a well-defined problem (the aim is clear). The difficultly lies in the open-nature of the classification that follows this lower-level process (which is discussed in the next section).

Classifying Aerial Datasets

There are two key types of classification that are used during semantic change detection. Firstly the classification of the type of a cluster of points. Secondly the classification of the type of variance present between two distinct clusters of points. I refer to them as type classification and variance classification respectively. In type classification the goal is to distinguish salient objects from clutter objects. Here that means identifying architectural objects and omitting organic and all other non-salient objects. In the case of variance classification the requirement is to identify and localise instances of pair-wise variance using a finite set of high-level

classes of change. Type-classification methods are reasonably well established and with the exception of considerations for input-data-type and sensing-noise, rely on calculating characteristic properties of an object-representation including:

- **Disparity, Regularity** is a measure of how disparate a point is relative to its neighbours or how regular a point is relative to its neighbours. In [166] this means considering $f_1 = \|p - \bar{p}\|$ to measure the distance between a point and the centroid of its neighbours. Regularity terms typically results in values close to zero for points on architectural surfaces. In [73] they consider instead a scatter term (f_s), defined as a local measure of height dispersion amongst neighbouring points.

- **Planarity** is a measure of the extent to which a point belongs to a planar (flat) surface. In [71, 73] this corresponds to a local-non-planarity term f_p which is derived from the quadratic distance to the optimal plane formed by a point and its neighbours. However in approaches such as [166, 164], they refer to it as a flatness term (or surface-variation)

$$f_3 = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}$$

  and calculate it via covariance analysis - where λ_i is the i^{th} eigenvalue of a point neighbourhood's covariance matrix (sorted in ascending order) such that $\lambda_0 \leq \lambda_1 \leq \lambda_2$. In both cases the smaller the response the greater the likelihood that the point represents a planar roof point.

- **Stability** is a ratio of the balance of disparity to regularity for a cluster of points. The return value is often a scalar in the interval [0:1].

- **Elevation** is a measure of a point's height above the ground terrain. Given the physical nature or urban environments, it is common for classifiers to consider how far a point is from the terrain and consider greater elevations as indicators of buildings. In the case of [72] this maps to their elevation-term f_e which uses a planimetric projected normalisation to estimate a points distance to the ground.

- **Curvature** measures seek to indicate whether a point belongs to a continuously varying curved surface. There are a handful of well-known curvature measures (some intrinsic - independent of embedding, others extrinsic - depending on embedding). For example the general Riemann curvature tensor [127] characterises curvature in n-dimensional spaces. In the case of surfaces in R3 the Gaussian curvature is also applicable [127]. For point-sets another strategy ([107],[88]) is to consider the rate-of change of a point neighbourhood's unit-normals on a Gaussian sphere. Although (as in our case) the return value is often a scalar, some measures return curvature vectors [127].

- **Directionality** is a measure based on the principal-axis or the normals of a set of points. In particular for aerial lidar the expectation is for points that correspond to roof surfaces to feature 'upward' pointing normals. In [167] they define this as a 'horizontality term' stated as:

$$f_2 = 1 - |n_p \cdot e_z|$$

  where $e_z = (0, 0, 1)$ is the vertical direction and n_p is the point's normal (which they obtain via covariance analysis). They use the convention that the z-axes

maps to the vertical up-direction. To derive point normals - their equation for solving the eigenvector problem is: [167]

$$C_p = \frac{1}{|N_p|} \sum_{q \in N_p} (q - \bar{p})(q - \bar{p})^T$$

The notion of directionality can also be used to filter points corresponding to walls for example if $f(n) = n_z - |n_x^2 + n_y^2|$ yields a measure of the angle between the points normal and the ground-terrain. Then the expression $f(n) < 0$ can be used to classify wall points by determining points on surfaces with greater vertical variance than horizontal.

- **Smoothness** a measure of the smoothness of a locale of points - considering the scope of perturbations in higher-order derivatives. Both planar and curved points respond to this discriminant feature. For example in [107] a smoothness term defined as $\|n_p \cdot n_s\| > \cos(\theta_{th})$ is not only used to classify point-types but also supports the subsequent segmentation. It computes the variance between neighbouring points n_s and n_p, as the magnitude of their dot-product, thresholding the result with a user-supplied maximum variant angle θ_{th}.

- **Extents** are measures of the span of the cluster of points under consideration (an axis-aligned bounding box - AABB or minimal-volume bounding box - Hough-transform). Extents apply to a set of points and are typically used to filter points by local neighbourhood or at the level of segmented objects. For architectural classification the intuitive logic is that greater volume corresponds to rigid man-made structures.

- **Variance, Spread, Distribution** are statistical measures of the scope of the 'scatter' present in a cluster of points using implicit geometric properties (elevation) or derived attributes such as surface normals. In particular a normal-distribution term plays a key role in classifying vegetation points in [162] [167]. They consider the solution to the eigenvector problem with a larger neighbourhood support radius in constructing a normal covariance matrix: $N_p^n = q | q \in P, \|p - q\| < \eta$:

$$C_p^n = \frac{1}{|N_p^\eta|} \sum_{q \in N_p^\eta} n_q^T \cdot n_q$$

where eigenvalues $\lambda_0^\eta \leq \lambda_1^\eta \leq \lambda_2^\eta$ have corresponding eigenvectors $v_0^\eta, v_1^\eta, v_2^\eta$. As Garland[43] and Pauly[102] pointed out, one can consider λ_1^n as a measure of normal variation - predicated on the fact it measures the maximum variation of normals about the Gaussian sphere. Hence the use of the feature $f_4 = \lambda_1^n$ in Zhou's 5-dimensional linear-classifier. Covariance analysis in-particular lends itself to other domain-specific feature detectors and another that Zhou exploits is the notion that a regular band on the Gaussian sphere corresponds to sharp features at the intersections of planar roof-patches. This heuristic feature enables their classification of roof-edge points (analogous to edge-detection).

- **Density** is a measure based on the scope and spacing of each point's neighbourhood. As an example in the point-cloud library [115], the density of a

point (coupled with other statistical measures) is exploited to identify and remove outliers from unstructured range-scans.

In terms of variance-classification in airborne scan datasets - (to the author's knowledge) there is not a direct analog to the classification of types of change represented by a cluster of deviant points in the pre-existing literature. As such this aspect of the semantic change detector constitutes a novel extension to conventional change-detection strategies in this domain.

In the course of presenting the approach the chapter refers back to the discriminative attributes and classification approaches in order to define the type-classifier and variance-classifier used to characterise the temporal changes between the input (out-dated) CAD model and (up-to-date) LiDAR scan.

The thing to remain aware of is that in the case of variance-detection binary-classification yields little semantic information. For example a binary change detector might have class labels : equivalent, non-equivalent - which although accurate (to some extent) fail to express the nature of the variance present between the input geometries. Basically although the formulation of type-classifiers is reasonably intuitive (determine the class of a single object), for variance classifiers (that determine the relationships between pairs of objects) it is critical that the return bear meaning within the domain.

This section briefly revised the relevant research in segmentation, change-detection and the classification of aerial point-sets. The discussion now progresses to the details of the approach taken to enable semantic change detection.

2.3 Methodology

The approach taken is designed to robustly classify multiple-modes of actionable temporal urban variation between CAD models and LiDAR scans.

Figure 2.5: *the chain of processes in semantic change detection*

Figure 2.5 provides a visual indication of the four key stages which involve:

1. Registering the out-dated CAD model to the newly acquired laser scan.
2. Segmenting the CAD model and laser-scan to identify salient objects.
3. Classifying the type of each segmented object and the class of variance present between its corresponding match in the alternate dataset.
4. Visualising and documenting the results of classification such that a human-operator can act upon them and an automatic algorithm can selectively re-construct only a subset of an up-to-date laser scan.

This portion of the chapter discusses each of the stages, detailing for each, the considerations and algorithmic steps undertaken. It begins with a complete overview of the change-detector, which is succeeded by the discussions of the registration, segmentation, classification and visualisation stages.

2.3.1 Outline

This section provides a brief outline of the body of the approach. It outlines the input and output data and provides high-level pseudo-code to clarify the change-detector's behaviour. For reference, figure 2.6 illustrates the semantic classes of (actionable) architectural variance that shall be introduced.

Inputs-Outputs

The semantic-change-detector takes as input: a DSM .asc file - a recent Digital-Surface-Model scan of the region, a DTM .asc file of the corresponding Digital-Terrain-Model, and a CAD file containing an out-dated representation of the region. It uses the following control-parameters (supplied as input arguments, by an end-user): a scalar value representing the minimum building height in meters, a scalar value for the minimum footprint surface-area in meters squared, a vector of unit-less scalar weights to control vegetation filtering, a scalar - the maximum mean point-to-plane variance distance between equivalent objects in the DSM and CAD inputs in meters, a scalar - the maximum normalised deviance ratio (zero to one), and a scalar - the normalised noise contribution tolerance (zero to one). It computes and returns the following outputs: a text-file containing the properties of buildings identified in the out-of-date CAD model - a sequential set of comma-separated values, a text-file containing the properties of buildings identified in the up-to-date LiDAR scan - a sequential set of comma-separated values, a text-file

containing identifiers for each pair-wise match (as integer tuples) and corresponding variance classifications labels, a colour-coded 2D vector map of the variation between the two registered inputs - with an optional rasterised variance image, and a colour-coded 3D mesh of the variation between the two registered inputs. Figure 2.6 illustrates the classes of variance.

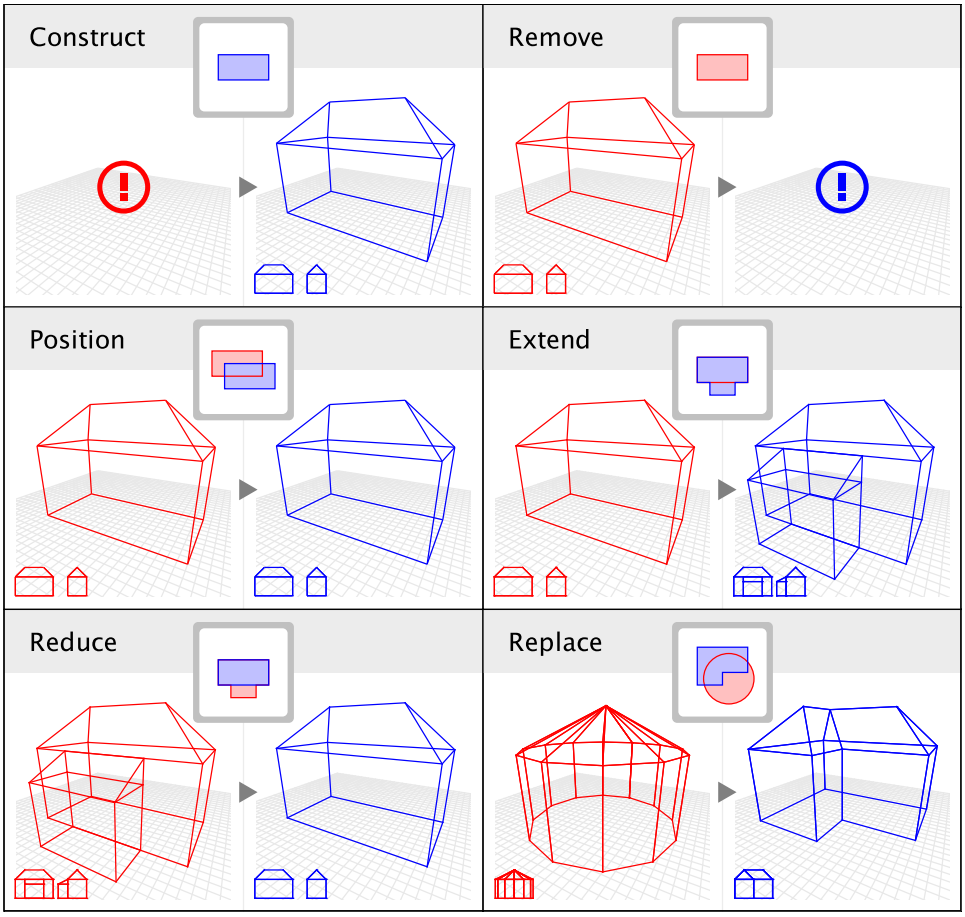

Figure 2.6: *the set of variance classes recognised by the semantic change detector - red elements represent the out-dated model, whilst blue elements represent the newly-acquired model.*

Each class corresponds to a distinct type of variation that the change detector should be capable of identifying. The first two (construct and remove) represent singletons (pairs of objects for which one of the pair is null), whilst the remaining four (position, extend, reduce and replace) require the out-dated object (red) be manipulated to match the up-to-date object (blue).

Although figure 2.6 illustrates both the old and new geometries using wireline rendered models, in practice the newly-acquired geometry is actually represented as a point-cloud, and a continuous representation must be derived from a set of sampled positions. The figure aims more to clarify the nature of each class of variance in a manner that is intuitive to understand.

Pseudo-Code

Figure 2.7 provides a clearer outline of the procedural steps employed to characterise the temporal changes between the out-of-date CAD model and the up-to-date laser scan. Each line represents an operation on the input data and they are grouped by the processing stage they belong to. Note the duality of the differencing, thresholding and connected-component operations that exists between the CAD and LiDAR during segmentation.

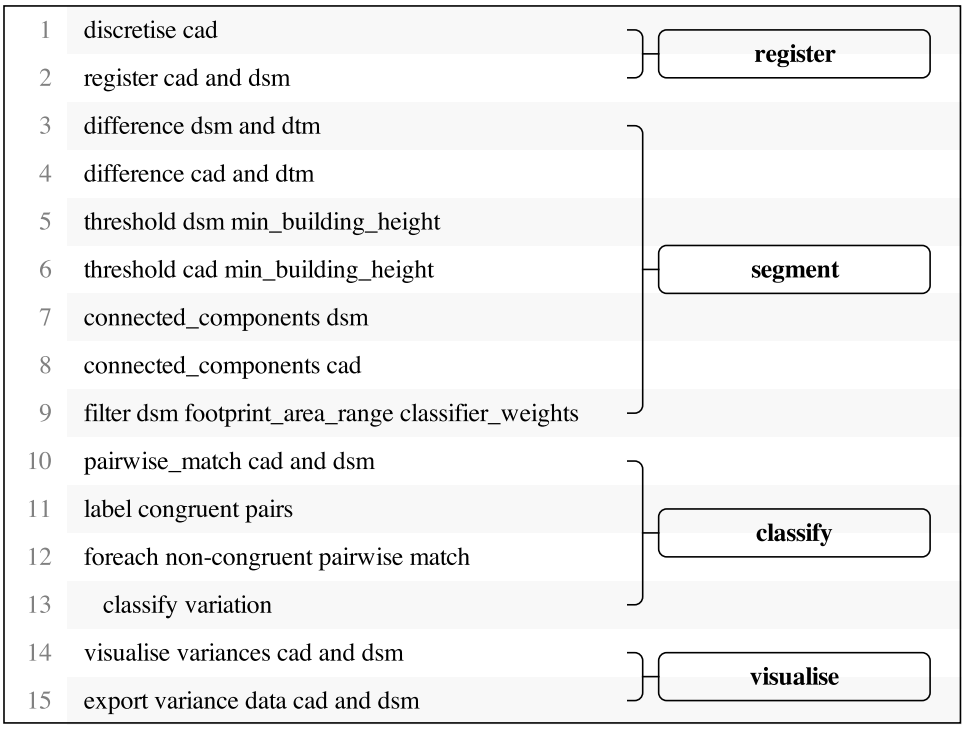

Figure 2.7: *high-level pseudo-code of the semantic change detector*

The pseudo-code aims to be abstract enough to work effectively on both structured and unstructured aerial laser scans. Although predominantly structured DEMs are used in the experiments, one should be aware that the same fundamental steps can also be applied to other classes of dataset. By omitting references to types, the pseudo-code provides a generic recipe for semantic change detection between multi-modal architectural assets.

Finally, before progressing to the body of the approach - the following figures depict the input data (the outdated CAD and newly-acquired laser-scan) that are used by the change-detector defined in this chapter.

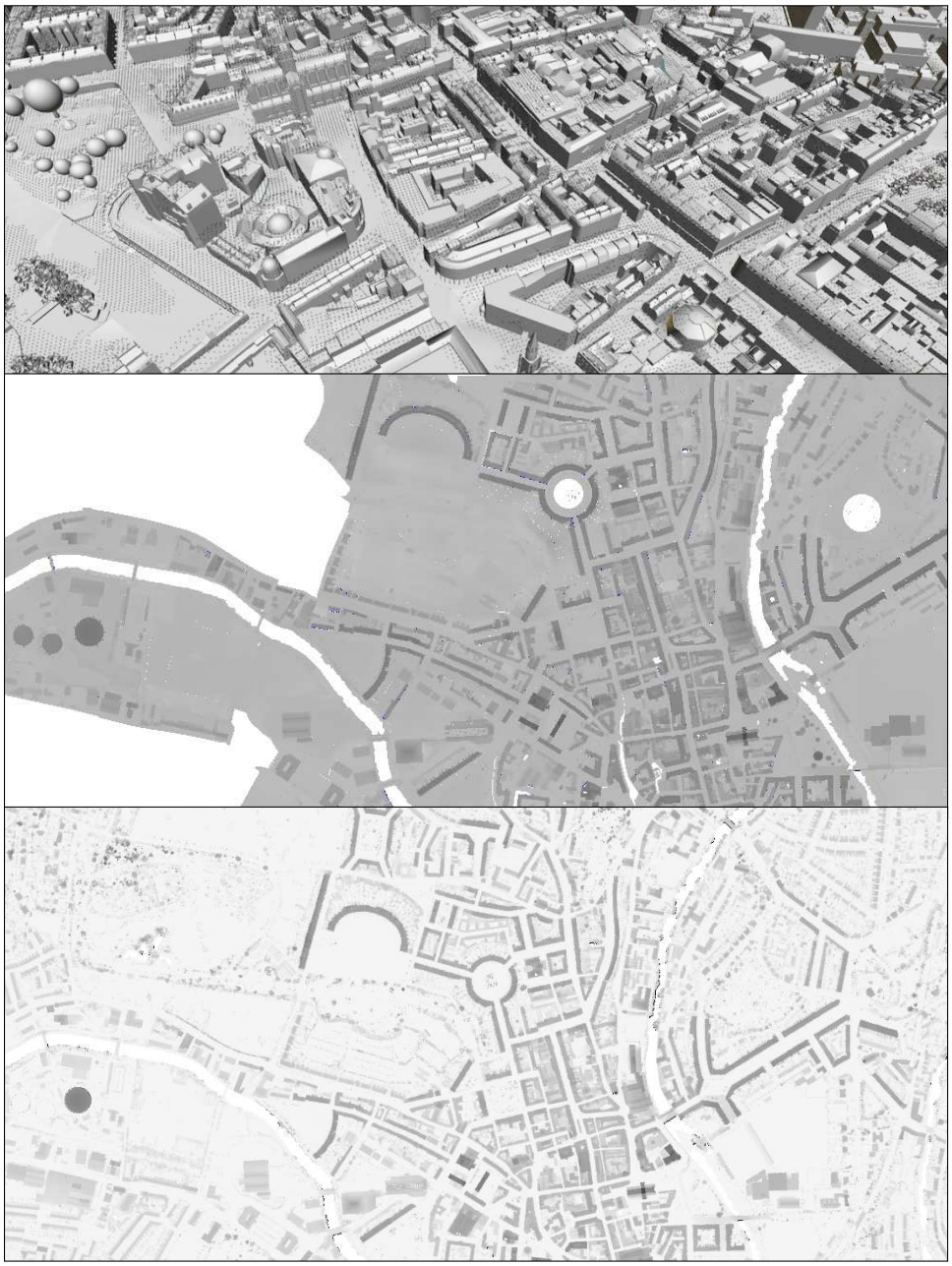

Figure 2.8: *CAD model (top), discrete CAD (mid) and 1m DSM (base)*

2.3.2 Registration

As discussed in part-I, there are a number of competing strategies, most notably for rigid geometries, ICP and Sliding-Window derivatives (with the latter being used heavily within image-processing [134]). In this case the aim is to align the out-dated CAD model to the newly-acquired laser scan. In order to achieve this one could perform multi-modal alignment by iteratively reducing the point-to-plane distance between the polygonal-faces of the CAD model and the scan points. However this form of registration relies upon the Euclidean distance metric - which is the most computationally expensive due to the square-roots involved. To limit this a sliding-window approach is proposed, which is akin to the techniques used in video tracking. The first step discretises the CAD model at the same resolution as the LiDAR points. Based on the parallax assumption (i.e. a 2.5D displacement field) [162], the method uses the Chebyshev distance metric since the expectation is that the deviation between corresponding points will occur along the y-axes (the vertical-difference). This resolves to the difference of the two range images (LiDAR and CAD) which can be computed in linear-time. The benefit of this is that it reduces the topology of the CAD to an equivalent representation as the LiDAR and one can also exploit the discretisation to support segmentation. The limitation of this approach is that the registration occurs over discrete data which introduces loss of information (which must be addressed). To combat this a sub-pixel registration refinement technique is employed, in order to limit the registration error to within a fraction of the discretising cell-size. The sub-pixel refinement routine is simply the same class of sliding-window error-minimising routine, but refactored to handle fractional offsets. In this manner the method first identifies an approximate registering transform and then refines the transform to achieve sub-pixel accuracy. Finally the registration method re-discretises the CAD model using the refactored transformation. The result of this process is a global rigid-body transformation which aligns the two input representations (CAD to LiDAR). Having outlined the method of registering the input geometries, the next topics detail the explicit stages and considerations. There are a handful of observations exploited to enable the sliding window routine to function as intended. They are stated below for reference.

- **North Coherency** of geographic and engineering representations. The north direction should be consistent between the CAD and the LiDAR - this enables the operator to omit the estimation of rotation during the process of registration. This principle was observed in the Bath dataset (CAD model and range-scans). Although there is no theoretical basis for it - generally speaking one can expect this to hold true - since the nature of the architectural and engineering domains typically requires that geometric assets of physical-sites can be geographically referenced, for which the convention is to have the north point upwards (+Y/+Z).

- **Scale Equivalence** of the discretised CAD relative to the input aerial LiDAR. The method exploits a uniform sample-step in order to negate the need to recover scale during registration. This is a product of the fact that both representations characterise the same physical region, simply in different mediums. Since physical-scale is a critical attribute of the both actively-sampled scans and manually-constructed models, the method expects 1 unit in the CAD to mirror 1 unit in the LiDAR. Again this principle was drawn from observation of the Bath dataset. Unlike north-coherency, this requirement is more likely to not be upheld by input data. In instances of geometric represen-

tations of the same object at different spatial scales, a registration function must also recover the scale component of the aligning transform. However in this instance, the principle holds and enables what is effectively a one-to-one mapping between points in the input laser scan and the discretised version of the CAD. In principle this principle need not hold true. All that is really required is that the algorithms be aware of the scale of both CAD and LiDAR - i.e. the mapping between world space and model space is known. This is because - by discretising the CAD model one can control the cell-spacing of the gridded CAD representation to ensure it matches the LiDAR.

- **Partial Correspondence** between the two input representations. The function expects that both LiDAR and CAD should represent the same physical region. This is critical - since without the existence of an equivalent region between the inputs the error-minimising transform returned by the sliding window routine will have little meaning relative to the human notion of alignment. In essence, part of the out-dated CAD and up-to-date LiDAR should be the roughly the same (subject to tolerance for noise). The size and location of the search window used during registration controls this - enabling the specification of a *congruent-region*. The key expectation of this constraint is that the scope of the registration error is small relative to the size of the input scans. Without this assumption there is no guarantee that the error-minimising transform will actually represent a coherent alignment.

These factors act as pre-conditions to the registration stage, and help ensure the correct execution of the alignment routine. One could exclude one or more of these constraints at the expense of having to use a more generic registration method, however due to the nature of the input data (used in this particular work) it seemed reasonable to take advantage of them to simplify the process as much as possible. The early-experiments with this approach determined that the only critical factor (that may affect the quality of the registration) is the level of sensing-noise present in the range-scan. However as long as the bounds of the stable window are large enough to enclose a set of (more than one) buildings, then the problem is largely mitigated since the larger the coverage the less likely a conflicting region (incorrect error-minimiser) will arise. The rest of this subsection provides a formal definition of the registration process in terms of its defining characteristics (the similarity measure, the transformation model and the optimisation method).

Formally the method seeks to find the transformation T that minimises the sum of the error squared between samples in the moving transformed object (CAD) - relative to the static reference object (LiDAR).

$$error(m, r, T) = \sum_{i=0}^{N} distance(T \times m_i, r(T \times m_i))^2 \tag{2.1}$$

where m is a reference to a set of sample points from the moving object and r is a function that returns the distance to the closest position on the reference object (given an input position). *distance(a,b)* calculates the magnitude of the difference between input vertex positions a and b. The expression $T \times m_i$ transforms the vertex m_i by matrix T, and could be equivalently expressed as $T(m_i)$. The transformation model is a rigid body translation - which is based on the aforementioned observations about the nature of the input data. The similarity measure used is the sum of the squared differences between the two dense matrices of surface

positions - which is stated as:

$$SSD(P,Q) = \sum_{i=0}^{N} (|\vec{p_i} - \vec{q_i}|)^2 \tag{2.2}$$

where: P and Q are the input geometries to compare (equivalent scale scalar elevation matrices), N is the cardinality of the input sets (the number of points), $\vec{p_i}$ and $\vec{q_i}$ are the ith elements of P and Q respectively.

The optimisation strategy is an exhaustive sliding window. At each optimisation step the SSD similarity measure is applied to a subset of the CAD and LiDAR to determine the suitability of the transformations $T_{[1-N]}$. The disparity minimising window that optimally aligns the discrete sets is:

$$arg\min(SSD(T_1 \times P, Q), \ldots, SSD(T_N \times P, Q)) \tag{2.3}$$

where: P is the point-set of the up-to-date laser-scan, Q is the quantized CAD model (a discretised elevation model), and the function $argmin()$ (argument minimum) returns the identifier of the transform with the least SSD error.

One particular point to note is the use of a squared error term in the similarity measure. One could conceive using simply the sum of absolute errors however the benefit of the squared error is the implicit penalisation to highly deviant vertex-positions. For example a single point with a variance of three units would contribute 3/n to the error term using the absolute sum and 9/n with the square distance. Further a point with variance result 0.5 units contributes 0.5/n to the error term using the absolute sum and 0.25/n with the squared distance. Essentially you can see the squared error term weights points with deviance distances less than one unit with less error and points with deviance greater than one unit with exponentially greater error. This observation can be used to normalise the scale of the inputs so as to further control the similarity measure. Although this is an interesting topic, it is somewhat tangential to our primary aims, so we conclude our discussion of registration here and move on to the task of identifying and isolating salient objects in the our newly-aligned CAD and LiDAR datasets.

To wrap up registration and to help clarify (before progressing to the discussion of segmentation) figure 2.9 illustrates the resulting aligning registration for the input CAD and LiDAR that composes the city of Bath dataset. The vital thing to remember is that registration using a sliding window (unlike ICP) is a deterministic (repeatable) process.

This determinism is key. Since all the subsequent processes are based on the initial alignment, ensuring that the operator can faithfully repeat the error minimising transformation, removes the potential for variability in performance that would be introduced at this early stage if a stochastic registration technique was used.

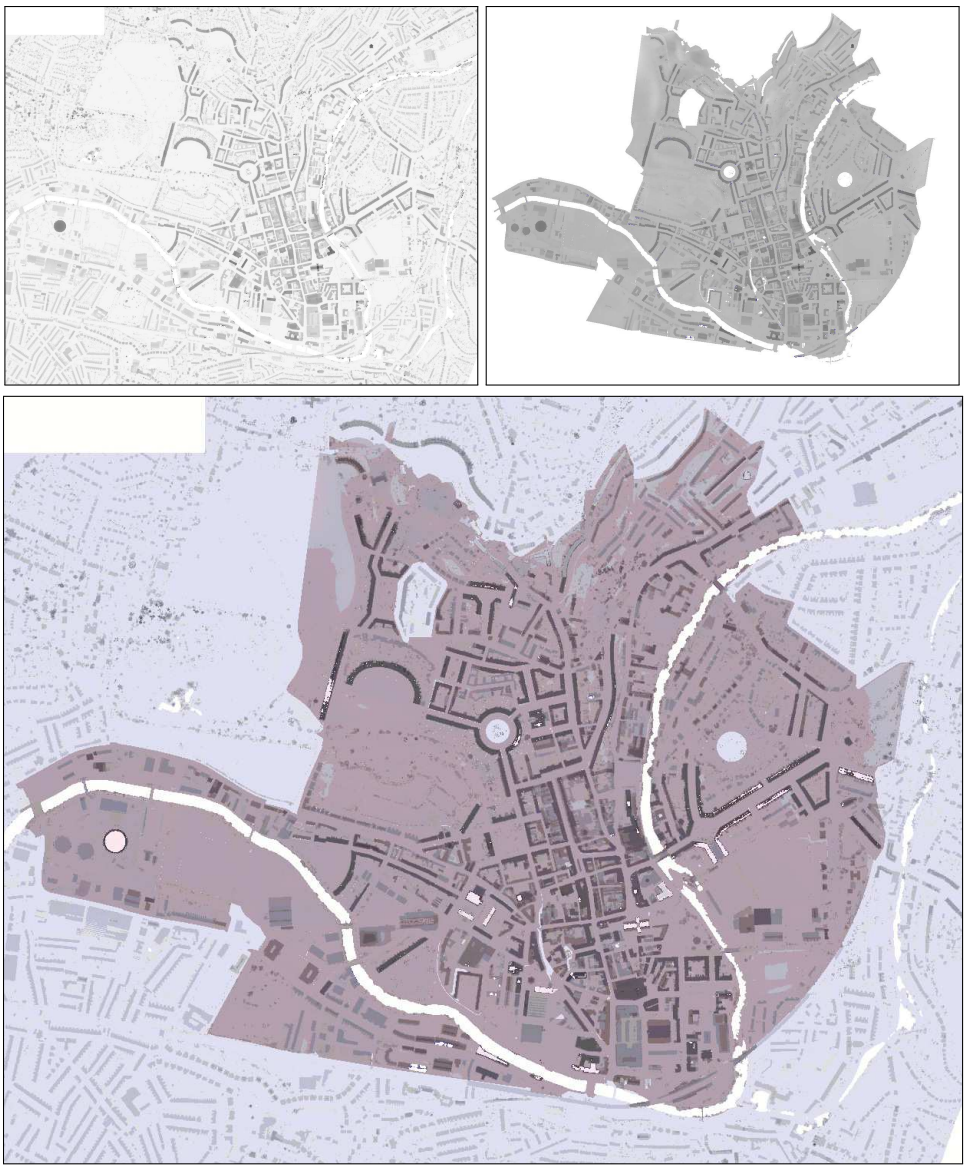

Figure 2.9: *registration of the discretised CAD and LiDAR DSM - CAD indicated by pale red and the DSM by pale blue - the top row illustrates the registered datasets separately*

With registration complete, the next section discusses the independent segmentation of the registered range-images.

2.3.3 Segmentation

The goal of segmentation is to divide something into its constituent parts [145]. Here the desired result is the extraction of individual building objects (descriptors of salient regions) from the registered CAD and LiDAR datasets.

Figure 2.10: *the chain of processes applied to segment the input datasets*

The semantic change detector segments buildings from a digital surface and terrain model using a 4-stage process. Abstractly (at a high-level) it:

1. **Slices** the surface model data into two sections (one containing salient clusters of points and the other containing terrain and clutter) - using the difference-of-elevation models principle (which is discussed next) and thresholding the result using a minimum-building height
2. **Dices** the stable set of clusters into individual building object descriptors by extracting connected components, after thresholding the difference-of-elevation models with a minimum building height
3. **Analyses** the isolated (segmented) building object descriptors, and for each computes characterising geometric properties (such as surface-normals, stability, disparity and variance) in a deterministic fashion
4. **Filters** the analysed building object descriptors - in order to remove any remaining organic, clutter and ambiguous objects - using an empirically constructed linear classifier - returning the remaining subset

Now that you have a high-level overview of the approach taken this section shall discuss each stage in the segmentation process in more detail.

Slice → Difference of Elevation Models

The approach discussed is based upon the difference of elevation models principle [12], [75]. Essentially the ground terrain acts as a coherent global reference between the datasets. Removing the terrain model from the surface model yields the difference of the elevation models. It is predicated on the assumption that the although architectural objects and organic elements change frequently, the evolution of the terrain occurs at a slower rate.

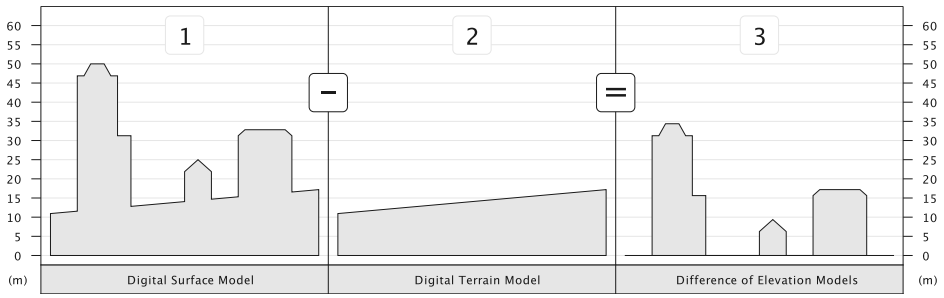

Figure 2.11: *the Difference of Elevation Models*

The method is conceptually simply and represents the slicing stage of the segmentation. Formally we can express it using the following expression:

$$DoEM = DSM - DTM \tag{2.4}$$

where $DoEM$, DSM and DTM are two dimensional scalar matrices of equivalent size - such that for each element $DoEM_i = DSM_i - DTM_i$.
An important consideration for the semantic-change-detector is the equivalence of the terrain model between the CAD and LiDAR inputs. The expectation is that the terrain acts as a constant that can be used for reference. As a result it is important that the CAD dataset models the variation in height due to terrain. However if the input CAD uses a uniform terrain model then the differencing should be skipped for the discretised CAD and registration should be performed with the DoEM as opposed to the DSM. For the datasets analysed in later sections, the difference of elevations is applied to the CAD and DSM models independently such that the result is two new elevation models. The final part of the slice operation is to threshold the result. For this simply exclude the points from each DoEM with elevation less than the minimum building height. This has the effect of removing many of the vehicles, small-trees and pieces of street-furniture that may be present as well as cleanly delineating connected components.

Another important consideration is the automatic derivation of digital terrain models (DTMs). Unlike the digital surface-models (DSMs) which are directly scanned samples of the physical surface, terrain models are generally derived from approximating the ground using a surface model and then physically surveying a selection of locations to confirm correctness. Essentially the algorithm anticipates the fact that the DTM (supplied as input) may be subject to greater geometric error than the DSM. Essentially DTMs are back-calculated from DSMs (generally by the scan provider) and as such there are no special hardware facilities exploited in acquiring them. Anybody with access to a DSM can also implement an equivalent DTM approximation method based solely on the DSM. This is key to ensuring that even in the case of a missing terrain model the operator is still be utilised.

The minimum-building height threshold aims to manage the error present in the DTM and essentially ensures that even in the presence of terrain estimation errors the operator can still identify salient objects. In the experiments discussed the minimum building height was set to 3.75 meters. Although this value can be altered, generally speaking it is best to use the minimum (smallest possible) height threshold (that omits the bulk of low-lying clutter objects), since a threshold that is too high will run the risk of failing to detect smaller one-storey buildings such as bungalows. Additionally since different minimum-building height thresholds will yield different slicing results, this value can be used to alter the behaviour of the segmentation stage. The useful aspect of this approach is that the threshold value is an intuitive physical measure (as opposed to a unit-less scalar). As such any sensible value will yield robust slicing results. As a guiding rule of thumb - values ranging from 1m to 10m provide a good starting point, from which you can refine the value to alter object recall and to better suit your particular dataset.

Dice → Connected Component Extraction

The next stage is to dice up each DoEM into sets of distinct buildings - each represented by a cluster of contiguous points. This stage is analogous to blob extraction in image processing and as such is treated in a similar manner. Figure 2.12 illustrates the logic employed to extract disjoint connected regions from the difference of elevation model masks constructed previously.

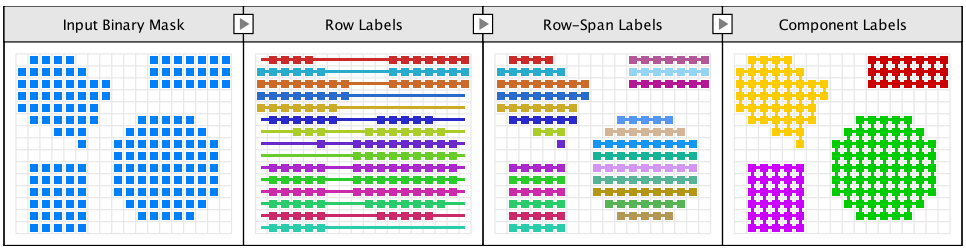

Figure 2.12: *scanline connected-component routine illustrated*

As connected-component extraction has a well established research history [134] full details of the routine are omitted. Rather the key aspects of the function are discussed. Formally the function takes as input a regularly spaced cartesian grid of boolean states and results in a integer label grid of equivalent size, such that each element in the result represents the identifier of the group that the element belongs to. It uses the convention that group zero corresponds to background and all other groups (with identifiers greater than or equal to one) represent the distinct components in the input binary mask. There are three common strategies typically used to achieve the result. These are seeded-flood-filling, scan-line-conversion and graph-traversal. For all, the labelling result should be consistent, however each approach bears difference computational performance characteristics. The approach taken by this work is scan line conversion, since it is significantly more efficient than flood-filling, whilst using less memory than graph traversal. The key steps in the scan-line connected-component routine are:

- Label Scan Rows - by trivially iterating over grid rows
- Label Spans in Scan Rows - based on linear disjointness
- Connect Adjacent Spans - (from top down) based on linear overlap

The method is conceptually simple and positively handles arbitrary shapes based solely on connectivity. The reason it works well for aerial laser scans of urban regions, is that the expectation is for buildings to be disjoint.

Readers familiar with the Mathematical environment MatLab should already be aware of the bwlabel() and bwconncomp() functions which provide results analogous to the outcome of the function described above. One slight difference between this implementation is that by default this employs 4-way connectivity logic in 2D as opposed 8-way logic. This alteration seeks to prevent diagonal weak-connections that may appear between distinct buildings (which are sampled at low resolutions) being propagated through to the subsequent processing stages. Although this is generally not necessary at point-spacings greater than 4ppm it proved helpful in limiting over-clustering (or under-segmentation) at 1ppm spacings and below.

Finally (having labelled each difference of elevation models threshold mask) the operator extracts groups of points by simply iterating over the grids and collating individual elements based on whether they share the same group identifier. This results in two sets of sets of points. Each set of points (in each set of sets) corresponds to the grid-locations of a connected component.

Having diced up the difference of elevation grids into sets of distinct connected components, the next processing stage is the analysis of each component - to

determine salient object properties. The next section discusses the discriminative attributes that are computed for each segmented object.

Analyse → Implicit and Derived Properties

This stage corresponds to the analysis portion of the segment approach.

Recall from the related work the attributes used to describe segmented objects by Lafarge et al. [73] and Zhou et al.[162] In a similar manner a set of implicit and derived properties are used to support the automatic categorisation of each connected cluster of segmented points (as salient or clutter).

The implicit properties of a segmented object include the extents of the object (a 2D bounding-box and 3D bounding-cuboid) the surface-area of its footprint, the volume of its mass and centroidal points. By considering the relationships between neighbouring points, a number of additional attributes are derived, that describe both characteristics of individual points and an object as a whole. These are an object's disparity, planarity and stability.

The disparity of an object is defined abstractly as a cumulative measure of the extent to which neighbouring sample positions on the segmented object's surface deviate from one another and formally is computed by:

$$disparity(vert) = \frac{1}{N}\sum_{i=1}^{N}\left(\left\|\sum_{n=1}^{8}(vert_i - neighbour_n)\right\|\right) \tag{2.5}$$

where: $vert$ is the set of clustered points, N is the total number of points in the cluster and $neighbours$ refers to each point's neighbourhood. The return is a positive, unit-less scalar. Note that an object's derived disparity measure should not be confused with the measure of disparity between two geometric objects or point-sets that is used during registration.

The key insight in this approach is the fact that the method exploits the regularly spaced cartesian grid representation to compute 8 neighbouring 'cells' for each point in constant-time. Unlike the operator's predecessors the calculation of each range-point's neighbourhood relies upon image-based 'pixel-connectivity'. This means there are only really two direct options: 4-way or 8-way neighbourhoods. Alternatively a 'pixel-window' could be used to extract larger point-neighbourhoods, however larger neighbourhoods also have the effect of 'smoothing' implicit properties across point locales.

The planarity measure determines if a neighbourhood of points represent a set of planar faces by considering the variance between the mean unit normal and the K-combination set of unit normals formed for each vertex and its neighbours. Samples that are locally planar return values close to zero.

$$planarity(vert) = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{n=1}^{\binom{9}{3}} D_{Normal}(\mu\Delta, \Delta_n)\right) \tag{2.6}$$

This measure was loosely inspired by the difference-of-normals operator but rather than considering a points response with different support radii, it considers the magnitude of the variance of the space of all potential normals formed from selecting 3 points at a time from each points neighbour set. In the above expression

the combinatorial binomial expansion $\binom{9}{3}$ implies that for each set of 9 points in a point's neighbourhood (8-neighbours and the point itself) the set of all unique combinations of three points are evaluated.

The stability measure denotes the ratio of stable sample positions to unstable sample positions on a segmented object's surface by thresholding the sum of local absolute height deviations for each point's neighbourhood.

$$stability(vert, \epsilon) = \frac{1}{N}\sum_{i=1}^{N}\left(\left(\sum_{n=1}^{8}\frac{(vert_i - neigh_n)}{8}\right)^2 \leq \epsilon\right) \tag{2.7}$$

Unlike the previous two attributes (which return scalars) the stability measure represents a binary response - denoting stable or non-stable. The binary response aims to address the requirement for tractability in the result.

You'll notice the similarity of the attribute names used across this work and its predecessors. Effectively it is attempting to characterise the same features of clusters of points, but in a manner better suited to structured range images. Notice though the omission of attributes that consider the density of a point since the discretisation step regularises the inputs to a grid with uniform density. Having computed these descriptive attributes, they can be used to automatically isolate clusters of segmented points that do not correspond to buildings, discussed next.

Filter → Filtering Non-Building Objects

Before progressing to the variance classification stage, it is useful to remove segmented objects that do not correspond to buildings in order to ensure that clutter elements such as vegetation, vehicles and street-furniture do not impact the subsequent analysis. The aim of filtering non-building objects is to ensure as much as possible that only coherent building masses are analysed and that erroneous responses are not triggered by organic and dynamic objects. Without this post-segmentation filtering-stage large clusters of trees would need to be manually omitted from an updated model after reconstruction. The critical thing to be aware of is that this form of classification isn't really open to a closed form analytic solution. The problem is that even at high resolutions, artefacts such as vegetation-overlapping-buildings and discontinuities due to sensing noise can cause clusters of points (assertable as architecture to a human) to be treated as vegetation and vice-versa. In this sense computer aren't as effective as human beings at this type of qualitative differentiation - and as such tree-detection algorithms by and large rely on heuristics supplied directly or through training as opposed to concrete abstract predicates. Hence the use of an empirically constructed linear-classifier to distinguish organic objects from architectural objects:

$$f(A, man, \vec{w}) = \frac{(w_1 A_D + w_2 A_P) \times (1 - A_S)}{w_3 A_A + w_4 A_V} < 1 \tag{2.8}$$

where: A_D is the object's disparity, A_P is the object's planarity, A_S is the object's stability, and A_A and A_V refer to the object's footprint area and volume respectively. The influence of each component is controlled by the weights w_{1-4} of the vector $\vec{w}$. Intuitively this means the greater the mass of an object the greater the likelihood it is manmade, whilst ensuring that as disparity increases and non-planarity dominates, the likelihood of an organic element rises. Inverting the stability measure

ensures that complex manmade objects that are curved or appear non-planar, yet exhibit overwhelming continuity, are not mistakenly treated as organic.

For the experiments discussed later on, suitable weights were derived by a process of iterative refinement. Note though that for resolutions of less than 1ppm such as 2m resolution DEMs the disparity weighting had to be factored to mitigate the increased proportion of noise. Positively though this method is largely independent of resolution. Since it exploits the scale of the models to compute integrals, the weights associated with the surface-areas help ensure analytic equivalence. For example one can increase the weights associated with the surface area and volume to force smaller objects to be treated as buildings safe in the knowledge that the alteration will yield the same classifier behaviour consistently as the resolution of the datasets increases and decreases. The negative side of the linear filter is it assumes a clear division between the organic and architectural class. As already discussed this may not alway be true. It also assumes uniformly distributed sampling noise - which also may not always be valid for some building instances. To address this, one must identify object's that are ambiguous (based on the object's properties and distance to the separating hyper-plane that distinguishes the two classes) and treat such cases as requiring human validation. This is like an epsilon check when computing whether a point lies on a plane. In essence this considers clusters of points that possess the both organic and architectural qualities as requiring manual consideration.

This section discussed the process of isolating individual building objects in the CAD and LiDAR, and you should have good idea of the key steps in pretty much any segmentation algorithm (slice, dice, analyse, filter). However we still lack knowledge of how the two datasets interrelate. The next stage is to analyse the building point-clusters to gleam this information.

Before progressing to classification of pairs of objects the final portion of this subsection seeks to clarify the key concepts and the inputs and outputs of the segmentation process visually. Figures 2.14 and 2.15 illustrate the outcome of applying the steps outlined to the city of Bath dataset.

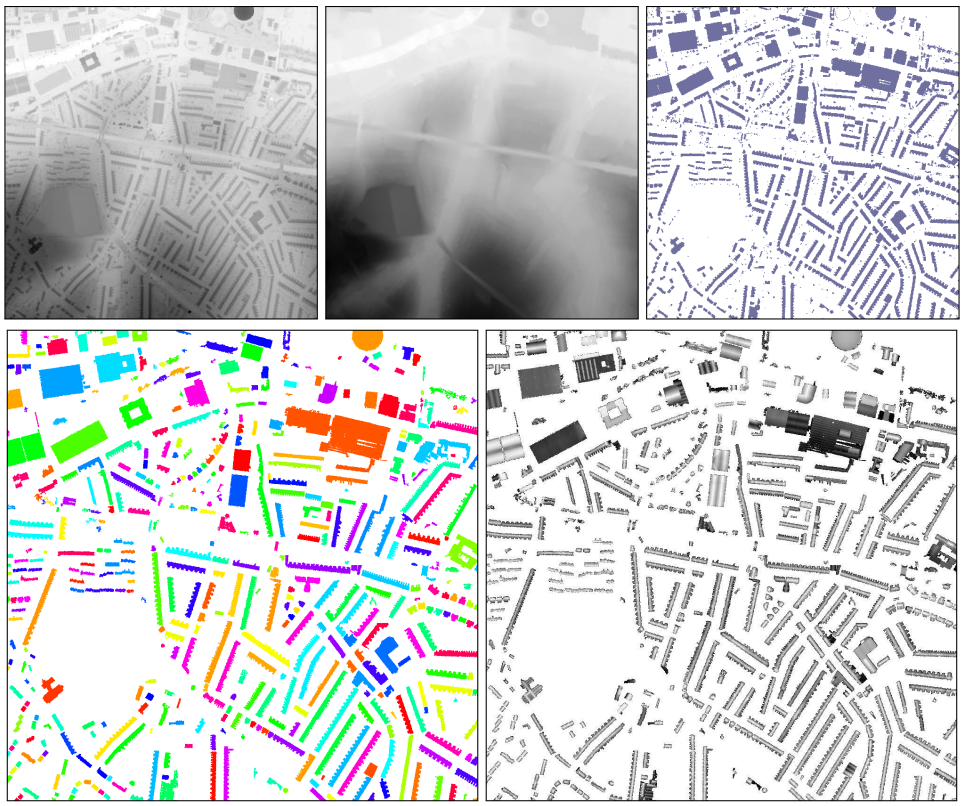

Figure 2.14: *The difference of elevation models segmentation (for a subset of the LiDAR scan) - from top left to base right: surface heights, terrain heights, threshold difference of elevation ($\geq$3.5m), labelled connected components filtered by area ($\geq 40m^2$) and finally the normalised object point clusters.*

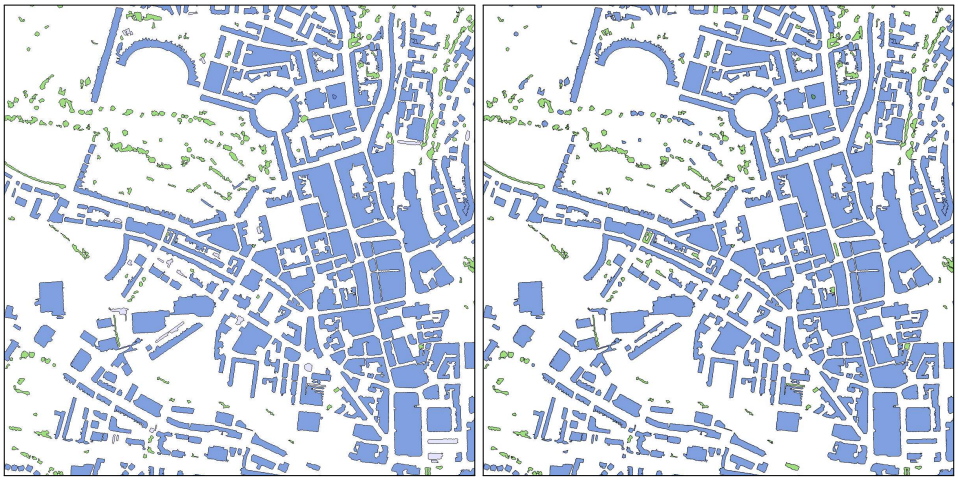

Figure 2.15: *Aerial-view of the ground-truth result (left) versus the classifier result (right) post manmade-organic classification for a subset of the LiDAR. Man-made elements are blue whilst organic elements are green.*

In summary, segmentation involves *slicing* each dataset into two distinct sets using the difference of elevation models, then *dicing* each salient set into individual connected components using a scan-line conversion routine, then *analysing* each remaining object to determine characteristic attributes and finally *filtering* vegetation and clutter objects using a linear classifier.

2.3.4 Classification

The result of segmentation is two sets of salient object clusters, one set derived from the out-dated CAD and the other from the newly-acquired LiDAR. Each object in each set caries implicit and derived properties computed post-instantiation. The next stage is to determine what needs to happen to bring each out-dated object into correspondence with its overlapping up-to-date object. So for every object in set A, one must first determine which object in B it interacts with and then categorise the type of variance between the two objects (in A and B). In order to achieve the first the requirement is a method of determining if two objects interact. For the second an ontology of the classes of variation of concern is needed.

The structure of the classification portion of this chapter is outlined as follows. First *pairwise analysis* is discussed, as a means to address the first requirement. Secondly variance-classification is considered as a means to categorise each instance of building-to-building variation present in the out-dated CAD model and up-to-date laser scan. During pairwise analysis the key objective is to efficiently match overlapping buildings and further to abstractly quantify the nature and scope of the interaction between them. In variance classification the aim is to label each pairwise match as corresponding to one of a finite set of potential *types-of-variance*.

For reference the types of variance are briefly described in the following enumeration. Additionally figure 2.16 illustrates 2D examples from the city of Bath dataset (at 1ppm) to clarify the appearance associated with each.

- **Congruent** - occurs whenever the variance between a building in the out-dated CAD and the newly-acquired LiDAR is negligible.
- **Construct** - operations correct instances of newly-erected buildings.
- **Remove** - operations correct instances of demolished buildings.
- **Position** - operations are needed to correct instances of building pairs that are geometrically equivalent but mis-aligned or out of position.
- **Extend** - operations are required to correct instances where a building representation has been extended - for example with a conservatory.
- **Reduce** - operations are equivalently required to correct instances where only a portion (or subset) of a building has been demolished.
- **Replace** - operations correct instances of highly varied buildings that cannot be effectively corrected with an alternative single operation.

Later discussions formalise these types, however it is important (useful) to be aware of them when considering the predicates exploited during pairwise-analysis - since it better contextualises the practical use of each predicate.

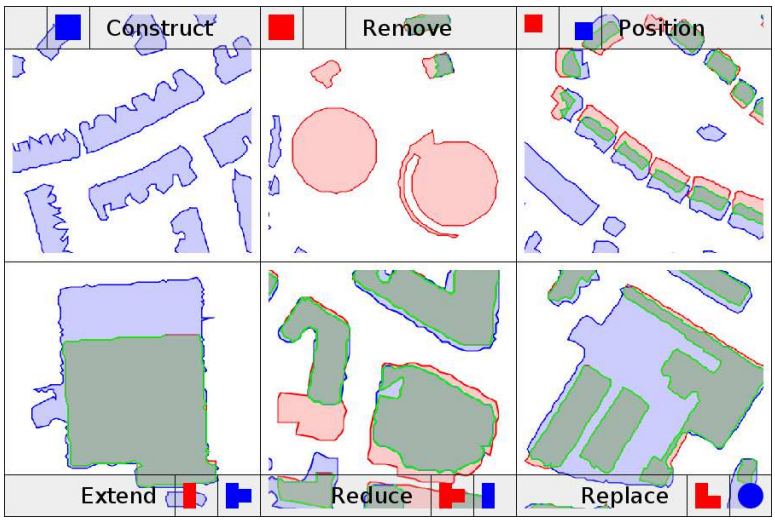

Figure 2.16: *examples of each class of geometric variation (for corresponding subsets of the CAD model and the LiDAR scan) - CAD in red, LiDAR in blue and regions resulting from pairwise object intersections overlaid in green*

Before detailing how each type of variance is detected the next sub-sections covers pairwise-analysis (which occurs prior to variance-classification).

Pairwise Analysis

An important precursor to execution of the detectors is pairwise analysis of the object-descriptors segmented from the CAD model and LiDAR points. Pairwise analysis involves two stages. The first is matching objects between the two models. The second is the deterministic instantiation of a set of constructive area and solid geometric regions. These derived geometries help to identify which class of variance is present for each pairwise match.

In order to match a segmented object in the CAD to its corresponding object in the LiDAR the operator exploits the notion of *overlap*. Basically objects are deemed corresponding matches if their geometries intersect. This applies to both two and three dimensional geometric representations. Hence the following predicate is used to test whether two building objects (one from the CAD and one from the LiDAR) are a pairwise match.

$$pairMatch(A,B) = \begin{cases} true & if\ (\int (A \cap B) > 0) \\ false & otherwise \end{cases} \tag{2.9}$$

This states that two objects A and B are to be considered pairwise matches if the integral (the surface area in 2D and the volume in 3D) of their intersection is greater than zero. This relies on the property of two non-interacting objects in space yielded null as their intersection. The only potential problem is the case where an object from either of the datasets, has more than one corresponding match in the alternate dataset. This can occur when a single building is removed and replaced by multiple smaller buildings or the inverse (many to one). An extension of this case is the presence of a many to many relationship between objects

in either one of the datasets.

Both of these edge cases represent instances of potential ambiguity that result from multiplicitous matches in either dataset. The issue is that the predicates discussed subsequently rely on a one-to-one mapping and as such both the one-to-many and many-to-many cases must be resolved. The operator exploits a simple strategy to address problems such as this.

Note though, that generally (in practice and in the experiments discussed later) the many-to-many relationship is rare, whilst the one-to-many (or the many-to-one) case is more common - as illustrated by the replacement in figure 2.16. However the resolution of both types of multiplicitous edge-case is handled using the same function - which implements the following rules:

1. If the number of overlapping matches for an object in the out-dated-CAD is greater than 1 : $\rightarrow$ then automatically remove the object in the CAD and replace it with the overlapping objects in the LiDAR
2. If the number of overlapping matches for an object in the newly-acquired-LiDAR is greater than 1 : $\rightarrow$ then automatically replace all the overlapping objects in the CAD with the single object from the LiDAR

These simple tests, enable the operator to filter out instances of buildings that would otherwise lead to erroneous classification results, and treat them consistently without the need for user intervention. However they represent the simplest solution to the problem and as such limit the scope of useful information that can be derived from corresponding problem case buildings.

With the ability to detect object-to-object interaction in hand, the next task is to compute derived geometries for each pairwise object match (CAD object and corresponding LiDAR object) - so to understand their interaction. This involves considering the derived set operator geometries in order to calculate a number of attributes of each match. They are used to characterise the scope and nature of the variation between the overlapping objects.

The positive and negative mass ratios indicate the distribution of change types over the non-congruent portions of the matched objects A and B. The result lies in the range zero to one inclusive, and simply measures the ratio of mass increasing change to mass decreasing change over the exclusive disjunction of A and B. The following expressions formalise this:

$$posMassRatio(A, B) = \frac{\int(B - A)}{\int(A \oplus B)} \tag{2.10}$$

$$negMassRatio(A, B) = 1 - posMassRatio(A, B) \tag{2.11}$$

These measures provide a means to quantify how much of the variance between two objects can be associated with an increase or decrease in mass. For example a high positive mass ratio (close to one) indicates that the bulk of the variation between objects A and B is the result of object B adding to the scope of object A. Inversely a high negative mass ratio (again close to one) between objects A and B means the majority of the variance between A and B can be attributed to the subtraction of geometry in B from A.

Further the following measure of deviance to congruence returns a normalised

scalar denoting what proportion of the union of A and B is outside of their intersection. This gives us a measure of congruency to change. The greater the return value the greater the proportion which is distinct. The lesser the return value the greater the similarity between the two objects.

$$devRatio(A, B) = \frac{\int(A \oplus B)}{\int(A \cap B)} \tag{2.12}$$

Note though that, unlike the previous two measures (whose return value lies in the range 0 to 1), the deviance ratio can yield positive values greater than one. For example if the scope (surface-area in 2D and volume in 3D) of the exclusive disjunction of objects A and B is greater than the scope of the intersection of objects A and B then $devRatio(A, B) > 1$. This is the basis for quantifying the portion of a pair-match that requires correcting.

Although this test is useful it does not indicate which class of deviation best describes the variance. It only indicates whether equality or variance dominates. For this reason the following test is introduced to determine the extent to which the deviation between the matched objects A and B can be characterised by a single type of change (namely as either purely mass increase or mass decrease). It considers the maximum of their mass ratio derivatives divided by their exclusive disjunction. For purely monotype changes (solely increase or solely decrease) the return value is 1, for perfectly balanced (half increase, half decrease) changes the return value is 0.5. Formally the monotype measure is given by the expression:

$$monotype(A, B) = \frac{\max(\int(A - B), \int(B - A))}{\int(A \oplus B)} \tag{2.13}$$

This construction yields values in the range [0.5,1.0]. To normalise it to the range [0,1]: the expression $2(m - 0.5)$ is applied to the result (m).

This is the key measure that enables the operator to distinguish between extensions, reductions and replacements. At its heart it indicates whether the variation between two objects can be corrected using one class of manipulation. In other words, it measures the balance between the addition of geometry and removal of geometry. If a variance is mono-modal then this implies that the bulk of the variation is attributable to one of either addition or subtraction. Intuitively this could also be inverted (1-result) to give a measure of how multi-modal each variance is - which would denote the requirement for multiple classes of alteration (addition and subtraction).

To reiterate the key concepts - pairwise-analysis involves matching corresponding segmented objects between the input CAD and LiDAR (based on whether their geometries overlap one another) and then constructing set operation geometries from the successfully paired matches. The derived geometries are used to determine what needs to happen to correct each out-dated object. However before we start classifying types of variations it is sensible to identify the congruent buildings in the input datasets. This requires a test for object equivalency, which is discussed in the next section.

Object Equivalency Testing

In order to determine if two corresponding pairwise-overlapping object matches are equivalent (which is an important aspect of differentiating congruent and de-

viant buildings) the following predicates are evaluated. The first is a normalised measure of similarity as a product of their overlap - the ratio of the integral of the intersection over the integral of the union. The second is a maximum average distance between corresponding sample positions on the two objects. In essence a continuous measure of geometric fit coupled with a discrete measure of the mean distance between their extremal boundaries.

$$equivalent(A, B, t, \epsilon)^{\mathbf{C}} = \left(\frac{\int ((A \times t) \cap B)}{\int ((A \times t) \cup B)} \right) \geq \epsilon \tag{2.14}$$

$$equivalent(A, B, t, d)^{\mathbf{D}} = mean_{err}((A \times t), B) < d \tag{2.15}$$

where *A* is the old CAD object, *B* is the new LiDAR object, *t* is an aligning transform, ϵ is the maximum normalised pairwise error and *d* is the maximum discrete deviant distance (the acceptable point-to-plane error).

The continuous equivalency test takes unit-less values for e_{max} in the range zero to one, whilst the explicit equivalency test accepts any unsigned real value to represent the magnitude of the d_{max} distance in meters. For the continuous test, objects that are in perfect correspondence yield normalised pairwise-error values of one, whilst deviant objects return values less than one, and non-interacting objects return zero. In the case of the explicit mean point-to-plane error, the tolerance depends upon the vertical accuracy of the point-set supplied as input. However sensible values typically fall in the range starting from the vertical accuracy distance through to two-or-three times the point-spacing across the lateral and longitudinal axes. The closer to zero the discrete limit is the fewer objects will be deemed equivalent in the presence of noise, the greater the limit the more forgiving the change detector will be of anomalous scan points and terrain estimation errors.

By combining the continuous and discrete predicate (using a bitwise AND) one can create a more robust equivalency test $equivalent(A, B, t, \epsilon, d)$ which considers both the interior interaction between objects and their extremal boundaries in order to determine congruency. This is applied to the pairwise matches to identify instances of buildings that can be included directly from the CAD. However for all the building-pairs that fail the equivalency test, one must iterate and determine the class of variation present for each. The next section discusses the classification of conflicting building pairs.

Detecting Types of Deviance

This sub-section deals with the detection of distinct types of variation. The input is a pair of overlapping building objects (one from the out-dated CAD, and the other from the newly-acquired laser-scan). The result is a class-label, that describes the nature of the variance present between the two objects.

Intelligent selection of deformation types is critical to an effective update scheme. The deformation types discussed were selected for their clarity and their intuitive mapping to the physical changes in urban environments. They aim to be succinct and complete whilst still yielding salient actionable descriptions of architectural change (that most importantly can actually be detected in the presence of sensing noise). One of the key requirements is knowledge of pairwise object matches discussed previously. Another is that the definition of each class of deviance is

mutually exclusive of the alternate classes. These constraints derive from the required behavioural characteristics set out in the overview section of this chapter. The driving desire is for each class of variance to bear semantic meaning from the perspective of a human CAD technician, such that the manipulations required to correct each out-dated building are evident in the results. The six classes of variance, detected by the operator (construct, remove, position, extend, reduce and replace) are not implicit and as such the operator must impose some formal logic in order to classify each. Simply put, the notion of an extension (as an example) is not a direct product of the geometry but rather a semantic label that only really bears meaning from the perspective of a human. It is a high-level descriptor used to characterise a particular type of alteration. Note though, that whilst the approach taken aims to formalise the abstract characteristics of each class, fundamentally the resulting logic is imposed explicitly and not derived empirically. This is important because it points to the fact that there are a myriad of other ways of imposing similar or distinct ontological expressions to sets of variance types. However you should also note the particular use of language. Linguistically each type of variance (detected by this operator) maps to a verb (a doing-word), that describes the operation required to correct the variance. This is not an accident, but rather an explicit attempt to maximise the utility of the information recovered by variance classification. The critical aspect of this is that surprisingly, both qualitative and quantitative variance types actually provide very little utility, when compared to discrete enumerations. To illustrate, consider the quantitative ontology of $\{$error $\leq$ max_error | error $>$ max_error$\}$, and the qualitative ontology of $\{$ excellent-fit | good-fit | ok-fit | bad-fit | terrible-fit $\}$. Neither of them actually describe anything of greater significance than a measure such as the RMS. In essence they fail to enable further manipulation based on the analysis, but rather represent a linear distinction. The point of all of this is that irrespective of the exact variance-classes you opt for, it is vital they correspond to *actions* as opposed to *measures*. This is because in order to use such a semantic change detector in a fully automated update mechanism, it is vital that the change-classifier inform a modelling algorithm as to *what* must be done, not *how much* needs to done.

Two parameters are used to control the operators tolerance of sensing noise, the maximum deviant distance d_{max} and the maximum normalised pairwise error e_{max}. They are analogous to those used during equivalency testing (part of pairwise analysis). In the equations and figures that are presented subsequently, the following conventions are observed. The symbol A refers to the CAD object and B refers to the LiDAR object. The symbol $\varnothing$ is used to refer to nullity (the null object), whilst the symbol $\oplus$ (as in $A \oplus B$) corresponds to the exclusive disjunction of A and B (which is analogous to the bitwise exclusive-or operation). The symbols $\cap$ and $\cup$ refer to the boolean intersection and union operations respectively. The integral symbol $\int$ denotes the surface-area (in 2D) and the volume (in 3D) or a geometric representation. The symbol t represents an error minimising aligning transformation, such that $A \times t$ corresponds to transforming object A by t. This could equivalently be expressed as $t(A)$ however the latter notation gives the impression that t is a function, whilst in reality it is actually a matrix.

Two additional figures are included to help contextualise each of the types of variance detected. The first (figure 2.17) depicts the logical hierarchy of the classes. Whilst the second (figure 2.18) provides concrete illustrations of the relationship between the derived set operator geometries and the characteristics of each class

of variance. Figure 2.18 is intended to be read from left-to-right so as to mirror the change detector's order of evaluation.

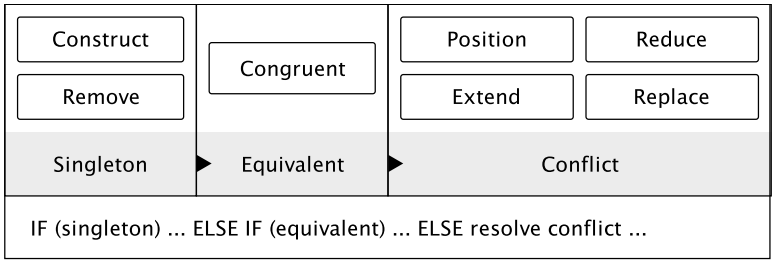

Figure 2.17: *the hierarchy of the detected variances classes*

One slight caveat. Even though logically the congruent class bears the highest discriminative level (since it distinguishes deviance from equivalence), in practice it makes more sense to evaluate singletons before searching for congruence. The reason behind this is discussed in greater depth shortly, however as a spoiler it basically boils down to the fact that the mere process of pairwise matching actually distinguishes (identifies) singleton variations simply by virtue of the presence of (or rather lack thereof) a match.

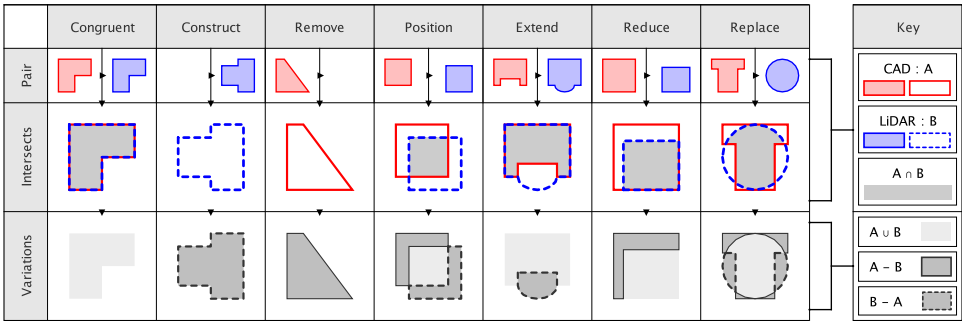

Figure 2.18: *diagram of the constructive set geometries for each class of deviance detected by the semantic change detector - the top row displays the input (out-dated in red, and up-to-date in blue) pairwise-overlapping shapes, the second the manner in which they interact (their common ground) and the third the type of variations present.*

Figure 2.18 indicates the core logic of this change detector. The key aspects to take note of are the variance characteristics of each of the classes. It illustrates noise-free, manually constructed geometries, for which the following high-level principles are observable in the table.

In instances of pure congruence, the geometries will be equivalent and hence the variations will be null. In instances of pure constructions and removals, one of the objects will be null and the variations will correspond (be equivalent) to the non-null entity. In instances of pure positionable variance, the geometries will be equivalent however the variations will be balanced (of equal scope) - half increase and half decrease. In instances of pure extensions, the geometries will be distinct, but only geometric addition will be present in the variations. In instances of pure reductions, the geometries will be distinct, but only geometric subtraction will be present in the variations. In instances of pure replacements, the geometries will be distinct, and the presence of geometric addition and subtraction will be arbitrarily distributed.

Critically the principles above use the term *pure* to refer to the absence of sensing noise and mis-alignments. Not the quality of the geometric representations.

These observations should be reasonably obvious. They are simply the product of considering the pairwise interactions between 2D and 3D geometries and isolating the characteristic (discriminative) features of each type of variance. The challenge however, lies not in the definition of these differences, but in enabling their robust identification in the presence of sensing noise. If the inputs were perfectly clean shapes and models, then this process would be trivial. In reality though the input CAD and LiDAR range images, are sampled from completely different mediums (one polygonal mesh, the other laser-scan), and possess very different noise properties. This means the change-detector has to be able to ignore insignificant deviances that may occur as a result of low and high-frequency noise, or the presence (or omission) of clutter found in the CAD or scanned in the LiDAR.

The next sections discuss the means of mapping these high-level observations to concrete predicates that can be efficiently implemented in a computer program. It details the formalism for detecting each type of variance.

Detecting Constructions

Construct - With the set of pairwise overlapping objects in hand, classifying an instance of deviance as requiring fresh construction is simple. If the result of pairwise matching for a point-cluster object (in the newly acquired LiDAR) is null (i.e. there is no corresponding object in the CAD model) then the algorithm has no choice but to construct a new object based solely on the LiDAR. Constructions are the result of partial coverage of the region within the CAD model and the result of newly erected buildings. The predicate following tests for the presence of a construction variation.

$$construct, \text{ if } (A = \varnothing) \ \& \ (B \neq \varnothing) \tag{2.16}$$

At a high-level this states that if the pre-existing object (A) is equal to null and the newly-acquired object (B) is not null then construct afresh a building from the newly-acquired object. You'll notice how simple this predicate is. This simplicity results from the fact that this class of variance doesn't have to consider the interactions between two geometries - it only has to determine whether or not their is a mapping between the datasets. Once one can guarantee that the newly-sampled building is completely omitted from the original CAD model then one can state unambiguously that the a fresh construction is required - without the need to interrogate further.

Detecting Removals

Remove - In a similar manner, remove operations are needed whenever an object cluster from the out-of-date CAD lacks a corresponding (partially overlapping) object cluster in the LiDAR model. The inverse of the construct operator - the need for this class of alteration is determined just as easily. If there is a CAD object with a null pairwise match in the LiDAR then the CAD object must be removed - since it is not present in the LiDAR. The key consideration is the potential for missing data in the LiDAR scan. However a simple means to constrain removal is to demand that there be points present in the LiDAR scan at the location even if they are below the

minimum building height used in segmentation. In this manner buildings are only omitted from the updated model if the algorithm can guarantee that the building no longer actually exists and isn't simply occluded or absent from the scan. The following predicate is used to identify removals.

$$remove, \text{ if } (A \neq \varnothing) \ \& \ (B = \varnothing) \tag{2.17}$$

This states that if the pre-existing object (A) is not null (i.e. there was a building present), and the newly-acquired object (B) is equal to null then remove the pre-existing building object. This predicate shares similarities with the construct class in that it can be detected unambiguously based on the presence of a sampled object and a null entity as its pairwise match. Next the detection of entities that require re-positioning is discussed.

Detecting Re-Positions

Position - The key identifying attribute of a position operation is that the topology of the objects under consideration should be almost identical (subject to tolerance for sensing noise). Hence to recognise an instance of a position (given two partially-overlapping object) one needs only to determine if by aligning the objects they become congruent. The transformation that registers the objects is also the input to the (subsequent automatic) re-position operation. This is stated formally with the following predicate.

$$\begin{cases} position, \text{ if } & (A \neq \varnothing) \ \& \ (B \neq \varnothing) \ \& \\ & equivalent(A, B, t, e_{max})^{\mathbf{C}} \ \& \\ & equivalent(A, B, t, d_{max})^{\mathbf{D}} \end{cases} \tag{2.18}$$

This is simultaneously the first of the variance classes that has to deal with noise and that operates on a valid pairwise match, since both constructs and removes can be detected irrespective of sensing-noise and represent singletons. The object equivalency routine discussed earlier is exploited in this instance to perform the same congruency test with a transformed geometry using a rigid body translation. To compute the translation, the operator simply reuses the same sliding-window registration routine defined in the earlier portion of this chapter, over the CAD and LiDAR object. This has vital implications since it means the operator is generally not invariant to rotations. Although minor rotational differences can be handled, significant mis-orientations between the CAD and LiDAR objects will generally limit the routines ability to identify a translation that yields congruence.

The thing to be aware of is that whilst this operator uses a simplified transformation model for detecting positionable variances, this is a result of the requirements for speed and determinism, and not because of a lack of alternatives. The reason that the input transform t is a matrix and not simply a vector describing an offset, is to allow alternative transformation models to be incorporated. How such transforms are computed is not for the classifier to say. For example 3-point registration could also be used as could ICP. The only problem with the latter is the introduction of stochastic iteration.

At this point the only remaining types of variance are extensions, reductions and replacements, and the operator can say definitively, that 1) neither of the objects is equal to null and 2) the geometries of the objects are distinct.

Detecting Extensions

Extend - The extension class aims to handle occasions when it is more efficient to append new geometric primitives directly onto a CAD object than it is to completely reconstruct the object using only LiDAR data. The characteristic difference between an extension (such as a conservatory) and a reconstruction (a demolition followed by fresh construction) is equivalent to the difference between appending and replacing. Namely the extension grows the original mass whilst ensuring that the intersection is equivalent to the original object. It is a mono-modal operation (the addition of geometry only) whilst replacing indicates that a mono-modal operation is insufficient to rectify the difference between the objects. The following predicate provides a formal definition of the logic exploited for extension detection.

$$\begin{cases} extend, \text{ if} & \left(\frac{\int (A-B)}{\int (A \oplus B} < e_{max}\right) \ \& \\ & \left(\frac{\int (B \oplus A)}{\int (A \cap B)} < 1\right) \end{cases} \tag{2.19}$$
$$\equiv (negMassRatio(A,B) < e_{max}) : (devRatio(A,B) < 1)$$

Intuitively you'll notice that this ensures that the normalised mass decreasing contribution to change is below emax, whilst also ensuring that the ratio of deviance to congruence is less than one. The impact of combining these two predicates is most apparent in the presence of noise - since it means that even with subtle variances (corresponding to reduced portions) the dominance of geometric addition controls the response of this class. Enforcing the deviance ratio also means that the operator will only treat a variance as an extension if the scope of the differences between the two objects is less than the scope of their congruent portions. In other words - it ensures that the amount of work associated with the geometric alteration maps to a fraction (and not a factor) of the overall out-dated geometry.

Detecting Reductions

Reduce - This operator is the opposite of extend. It encompasses the same challenges but yields the inverse result - the reduction of mass and a pairwise intersection that is equivalent to the new instance. The reduce operation manifests as the decimation of a subset of one of the out-of-date CAD objects. As such identifying a reduction requires a valid pairwise match (which is guaranteed after positionables are evaluated) and a decrease in the mass of an object as well as being a mono- modal instance of change (the removal of geometry only). The following expression formalised this.

$$\begin{cases} reduce, \text{ if} & \left(\frac{\int (B-A)}{\int (A \oplus B} < e_{max}\right) \ \& \\ & \left(\frac{\int (B \oplus A)}{\int (A \cap B)} < 1\right) \end{cases} \tag{2.20}$$
$$\equiv (posMassRatio(A,B) < e_{max}) : (devRatio(A,B) < 1)$$

This test is almost identical to the previous test for extensions, the only difference being the ordering of the input to the numerator in the first statement. It ensures that the mass increasing contribution to change is below emax (the normalised

continuous error threshold), and enforces the deviance to congruence check (described in the previous discussion of extensions).

By this point, five of the six type of variance can be detected, and the only remaining type is replacements. For this one could arguably use:

$$replace \text{ if } !(congruent|construct|remove|position|extend|reduce)$$

which, (although semantically correct), is needlessly inefficient from a computational perspective, since it demands each previous class be evaluated to identify replacements. The problem with this is that it will effectively double the execution time of the change-detector. It also means that if an end-user wanted to detect only replacements, they would basically have to wait for the full set of classes to be evaluated - even though only one is of concern. This is unacceptable, and so the final portion of this section defines the class of replacements independently of the previously defined classes.

Note that in the case of detecting all of the variance classes, the expression above could also be constructed by simply caching the result of testing each preceding class. However (although this removes the extra compute time associated with replace detection) it still frames the replacement type in terms of the other types, which will prevent further optimisations - such as culling the classification stage early when it becomes apparent an object must be replaced. For a practical example of this, refer to the implementation of the DEV algorithm (line 36), found in the appendix of this chapter.

Detecting Replacements

Replace - The replace operations are simultaneously the most powerful (in terms of raw error minimisation) and potentially the most expensive (in terms of computational workload). In principle every valid pairwise-match that fails the object equivalency test, is a candidate for the replace operator. Whilst this is might be acceptable for noise-free synthetic scenes, in practice this is undesirable, because sensing noise may distort the appearance of building objects that (to a human) are correctable using a simple change operator without resorting to a complete reconstruction. Hence determining the right tolerance for the object equivalency testing is critical in order to prevent overuse of the replace operator. If the object comparator is too strict the operator will favour replacing whenever it encounters deviation, however if the tolerance for error is too loose it will treat instances that actually require replacing as other classes of deformation (or worse fail to categorise them as varying). With this in mind, the following predicate is applied to determine the need to completely replace an out-dated building.

$$\begin{cases} replace \quad \text{if } \left(\dfrac{\int (A \oplus B)}{\int (A \cap B)} \geq 1\right) \; || \\ \qquad\qquad \left(\dfrac{max(\int (A-B), \int (B-A))}{\int (A \oplus B)} \leq \left(1 - \dfrac{e_{max}}{2}\right)\right) \end{cases} \tag{2.21}$$
$$\equiv (devRatio(A,B) \geq 1) | (monotype(A,B) \leq (1 - e_{max}/2))$$

Intuitively this uses the deviance to congruence measure and the monotype measure to define the class of replacements. If there is greater deviance than congruence (XOR over inter- section) then the object must be replaced - as is the case

if the variation exhibits as multiple modes of change (beyond the acceptable noise induced normalised deviances for extend and reduce).

To summarise the variance classification stage, this section discussed pairwise analysis as a means of matching and differencing overlapping objects. It covered the considerations of each recognised class of architectural change. It detailed the functions responsible for identifying buildings that need constructing, removing, repositioning, extending, reducing and replacing. Remember each detector function relies upon the boolean operations (encountered earlier) and simply applies them to the task of characterising variance in a robust (topologically-independent) way. It discussed the methods of mitigating noise and controlling the sensitivity of the replacement detector. It outlined how to resolve multiplicities in the initial object-matching stage and explained in detail the links between the variance classes defined and the properties of the semantic change detector sought. The result of all of this is a method of analysing temporally variant CAD and LiDAR datasets and finding the differences between then at the level of individual objects.

At this point one can automate the quantitative analysis of the change detector (by manually identifying exemplar variances in a test dataset and seeing if the detector correctly labels them with the associated class). However the ability to undertake qualitative evaluation is still limited. To address this a means to display and inspect the results of the classifier is required.

2.3.5 Visualisation

The aim of visualisation is to present the classified variation data in a manner that is intuitive for a human to understand and act upon. In the case of a fully-automated algorithm visualisation of variance between the CAD and LiDAR would be replaced by the correction of each variation. For a manual update the visualisation and documentation of the deviances represents the terminating return of the semantic change detector. There are a handful of key considerations which are enumerated visually for reference (in figure 2.19).

Topology	Dimension	Interaction	Shading	Integration
Points	2D	Static	CAD	Stand-Alone
Lines	3D	Dynamic	NPR	Plug-In
Polygons	4D		Photo-Real	Library

Figure 2.19: *table of the options for visualising classified variance data*

Having considered the available options for visualising the results of variance classification the proposed approach can now be outlined. This thesis exploits a stand-alone 2D engineering-document styled user-interface to display the classifier data. Figure 2.20 illustrates this. The program is simple and serves as an auxiliary reference for a human CAD technician that can be used alongside their preferred CAD environment. Additionally for debugging and command-line environments, 2D and 3D geometric data is written to disc such that an alternate viewer may also be used. A benefit of the proposed system is that it also provides a intuitive interactive dialog for the manual creation of ground-truth label data (figure 2.25) - which

is utilised later on to help quantify the operators performance.

These figures demonstrate aspects of the visualisation strategy employed. For each, the comments draw attention to the vital features depicted.

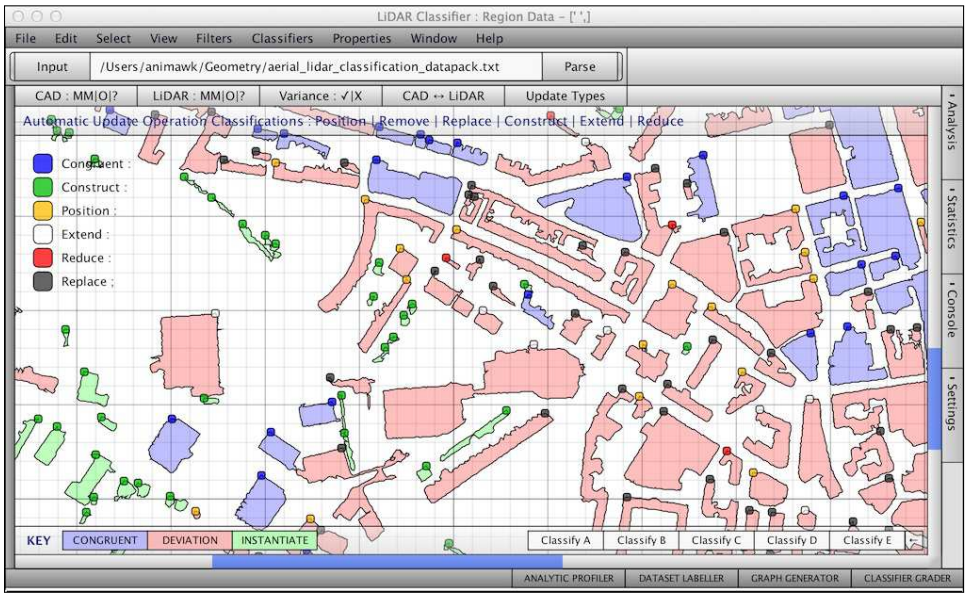

Figure 2.20: *result of classifying the update operations required to minimise geometric error for the City of Bath dataset - displayed in a simple UI - the maximum mean deviant distant between pair-wise object matches is 2 meters and the maximum normalised multi-modal error threshold is set at 0.1.*

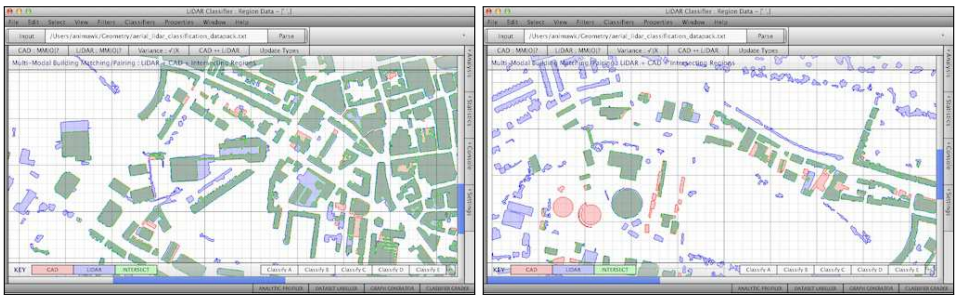

Figure 2.21: *visualising segmented objects in the 2D viewer program - out-dated CAD model objects are indicated by red boundaries, whilst newly-acquired objects from the Li-DAR DSM are indicated by blue boundaries - pairwise intersections are indicated by the overlain green boundaries.*

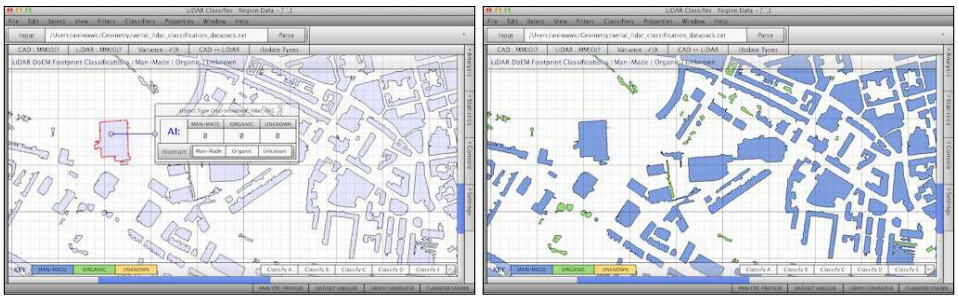

Figure 2.22: *visualising type-classification for the DSM of the city of Bath - left: the input segmented objects - right: automatic classification labels - with blue mapping to man-made elements and green to organic elements.*

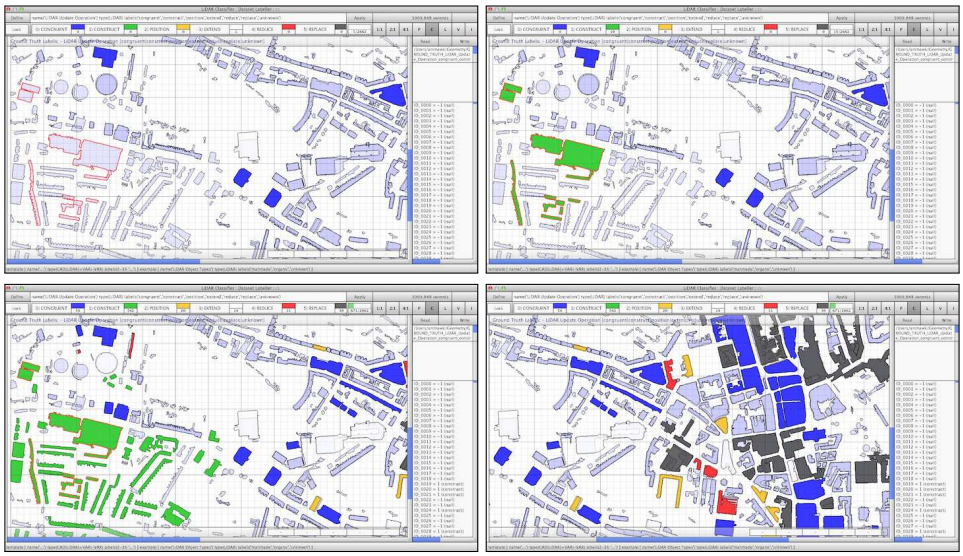

Figure 2.25: *Auxiliary dialog that enables end-users to create labelled ground truth datasets for quantifying the performance of the automatic classifier relative to a human. Users simply click on an object (to select and again to deselect) and then press (click) the corresponding variance class from the title-bar to assign the class of variance to the selected object(s). Users can also select multiple objects simultaneously using a drag box (rectangular region) selection. The program creates a list of labels for each object (right-pane) which the user can write to disk for use in quantitative analysis relative to the automatic classifier. Users can also load in previously saved sets of labels to make modifications. The title-bar buttons control the display of the input-data as points, objects or variances and holds a timer such that users can monitor the amount of time spent creating ground truth data.*

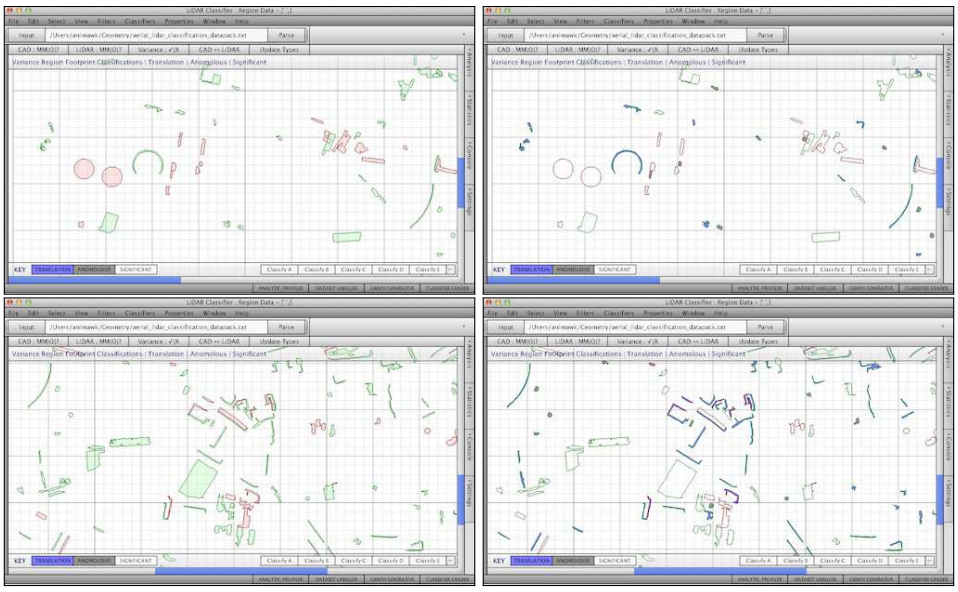

Figure 2.23: *visualising deviant regions between pairwise overlapping objects in the CAD and LiDAR of the city of Bath - left: segmented deviant regions with green mapping to geometric addition and red to geometric removal - right: automatically labelling each deviant region as either - induced by translation (blue), anomalous (gray) or significant (white).*

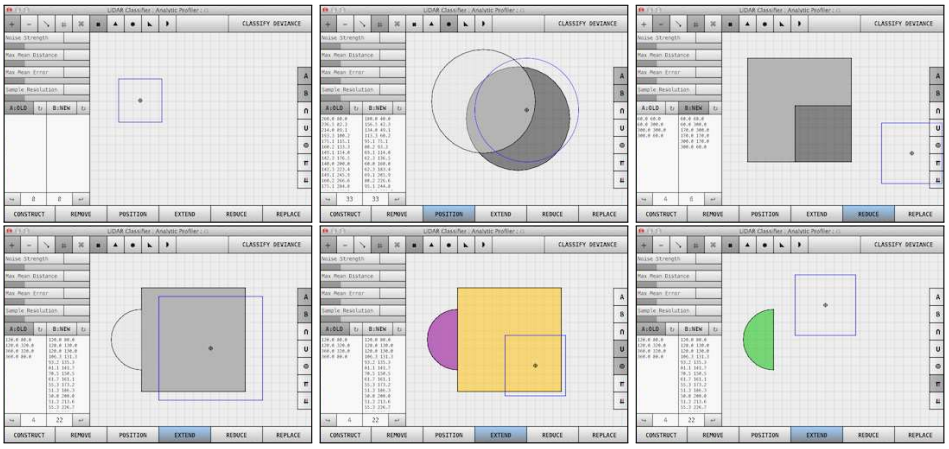

Figure 2.24: *Testing the classifier's logic in an abstract 2D program. Users plot (as input) two simple shapes using basic polygonal primitives and the boolean composition operators (CAG). Users can transform the shapes, apply pseudo-random-noise and vary the discrete and continuous error tolerance. The program analyses the input shapes, determines the type of variance present and highlights the corresponding class indicator (base-bar). The top row illustrates (from left to right) a new empty scene, a position-operation being detected and, a reduce operation being detected. The second row illustrates the auxiliary data that an end-user can inspect for the classification of an extend operation. It depicts (from left to right) the two input plotted shapes, their union (in goldenrod) with their exclusive disjunction (in purple) and, the scope-increasing contribution to change (in green).*

2.4 Experimental Results

This section presents quantitative and qualitative results of applying the variance classifier to different types of pairwise deviant geometry.

The goal of this section is to provide a clearer understanding of the physical performance of the semantic change detector on both concrete and synthesised datasets. The commentary in this section is minimal since the bulk of the high-level insights are discussed in the following analysis and evaluation section.

The key evaluataive attributes are (in terms of accuracy) the precision and recall (of the operator's class-labelling relative to human labelled data), and (in terms of efficiency) the growth in execution-time for each processing stage.

(Notes about terminology) In the tables that accompany the City of Bath results:- *the ground-truth count* refers to the number of human labelled instances of each class in the labelled datasets. The *classifier total count* refers to the total number of times the classifier fires for each class across the unlabelled data-set. The *recall correct count* and *recall correct rate* refer to the ratio of correctly identified classifier instances relative to the ground-truth data. The *true positive count* and *rate* refer to the ratio of correctly labelled classifier instances relative to the total number of times the classifier fires for each type. The *false positive count* and *rate* refer to the inverse metric - the proportion of incorrectly labelled classifier instances for each class. In the precision recall graphs the y-axes refers to the precision of each classifier whilst the x-axes refers to the recall.

2.4.1 Synthetic Datasets

This subsection briefly discusses the outcomes of performance analysis of the semantic change detector in controlled tests on synthetic data - for which the expected outcome was known ahead of time. These experiments enabled more thorough examination of the detector's behaviour than the real-world experiments allowed. In particular they allowed the dimension and continuity/discreteness of the representations used during analysis to be varied. The primary aim of these experiments was to determine the extent to which the processing stages employed by DEV generalise (adapt) to various types of building representation.

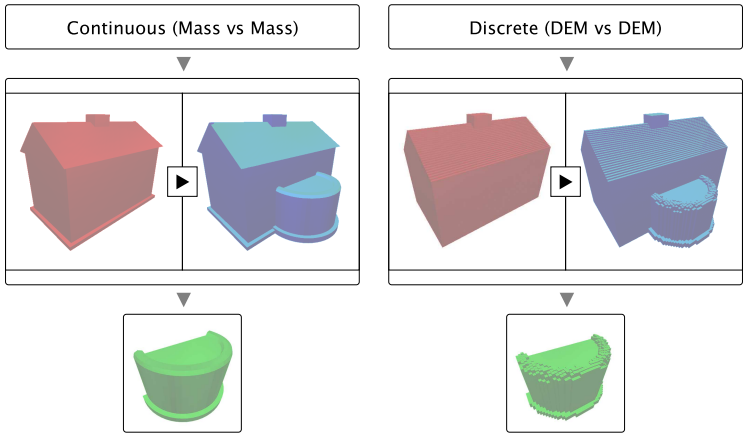

Figure 2.34: *the two types of 2.5D (top-down/parallax) synthetic test-case used during examination of the behaviour of the the semantic change detector - illustrating (left) polyhedral change-detection, and (right) discretised height-map change-detection*

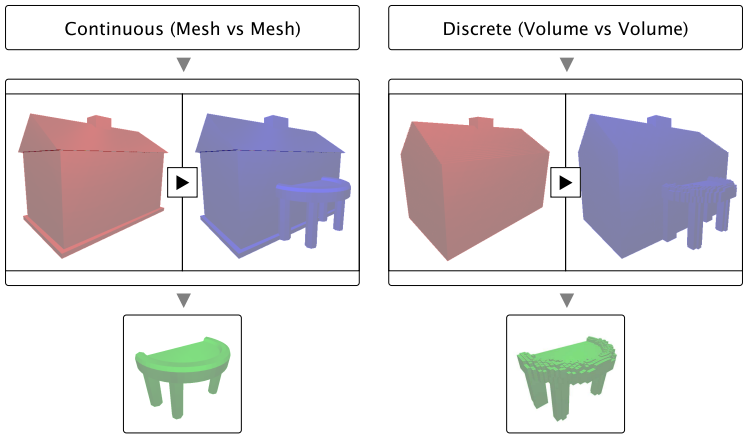

Figure 2.35: *the two types of 3D synthetic test-case for the semantic change detector - illustrating (left) polyhedral change-detection and (right) discretised volumetric change-detection*

The key observation in these controlled tests was the suitability of the change detection to various types of building representation. Indeed an unexpected outcome was the generality of the logic (its ability to adapt to 2.5D and 3D datasets) and in-particular the cleanliness of the variance regions when applied to mesh vs mesh change detection. This suggests that the proposed logic is also suited to mono-modal change-detection tasks on manually constructed CAD models. However as this is somewhat tangential to the primary use-case of DEV we shall leave the outline of the controlled experiments there and move on to the concrete results.

Note: the experiments on synthetic datasets do not form part of the concrete results for DEV. They simply helped to understand the limits of the approach during development and debugging. They are referred to here mainly for completeness.

2.4.2 City of Bath : 2D, 2.5D

This sections presents the results of performance analysis of the semantic-change-detector using the City-of-Bath dataset. The City-of-Bath dataset is composed of an out-dated CAD model, and up-to-date DSM and DTM range images. The out-dated CAD model was provided by the department of Architecture and Civil Engineering at the University of Bath and models the buildings, terrain and auxiliary features of the city center. Figure 2.8 illustrates the level of detail present in the out-dated CAD model and its discretisation as a 1m airborne DEM. The airborne DSM and DTM - which cover a 2.5km x 2km region of the city centre at 1m resolution (1ppm) - are composite DEMs produced by the Geomatics[47] group and provided as free to use research assets by the Environment's Agency [47]. Both the CAD model and the DSM and DTM scans use the OSGB36 coordinate system and are reasonably well sampled. In particular (with the exception of missing data where there are rivers and waterways) the scans exhibit good coverage of the City of Bath. To quantify the performance of the classifier, manually labelled versions of the data were also created to represent the ground-truth (or human-desired result) for the tasks of type classification and variance classification.

Three key aspects are considered, firstly the geometric correctness of the change-detector as a product of the reduction in geometric error that is achieved by correcting each building using the variance class labels. The second aspect is the performance of the type classifier that filters buildings from vegetation and clutter based on its detection precision and recall relative to human labels. Thirdly the performance of the variance labels, relative to human labelled ground-truth change-class labels. The first test considers how varying the pairwise error tolerance affects the resultant accuracy of an updated CAD model. The second and the third tests quantify the correctness and effectiveness of the open (ill-defined) classification stages.

Reduction in Geometric Error

The overall resulting reduction in geometric error is characterised below. It is a plot of the mean discrete point error (y-axis) for varying input values of maximum normalised continuous pairwise error tolerance (e_{max}, x-axis). The graph illustrates the global reduction in error as a result of correcting all classes of variance detected by the operator. To achieve this regular gridded chunks were removed and added to the CAD model automatically based on the detected errors. The individual point positions and elevations of each alteration were derived from the classified deviant regions. This provided an error minimising method of prototyping the actual update operations for each class of variance. Note however the geometry used represents the discretised differences and not continuous building model meshes. In essence this displays the new error between the CAD and the LiDAR if for each building variation, the deviant points are added or removed using a primitive dense mesh. The approach removes additional error that would be introduced by approximation and forecasts the idealised enhancement if unaltered versions of the detected deviant regions are used to drive the correction process.

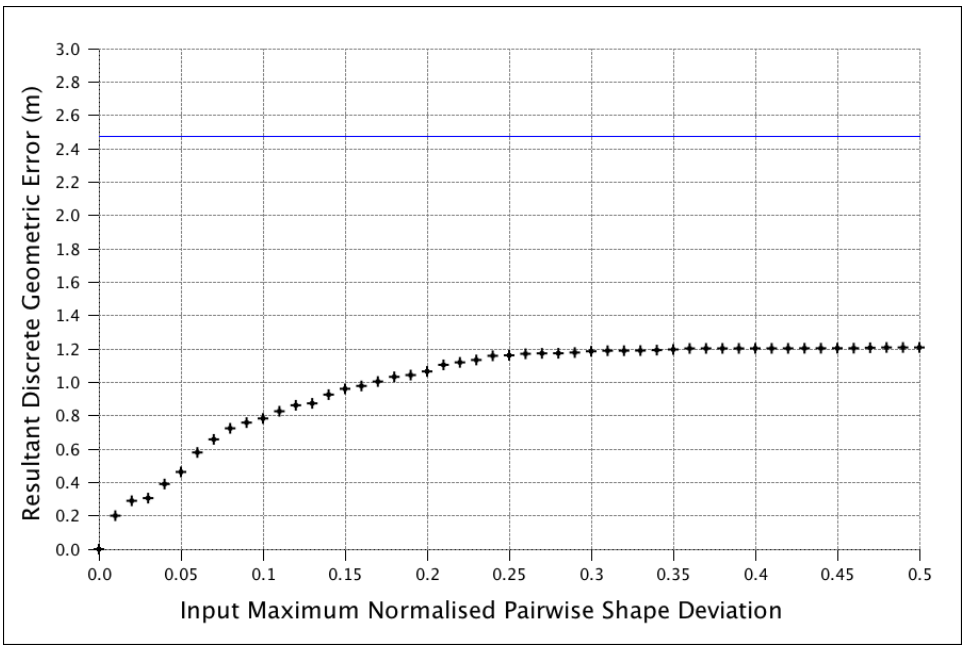

Figure 2.26: *graph of the resultant mean discrete geometric error in meters (y-axis) against max normalised continuous pairwise error (x-axis)*

The blue line is the original error between the CAD model and the LiDAR. The graph shows clearly that as the input allowable pairwise-shape deviance tolerance rises, the reduction in geometric error decreases. This makes sense since if two objects are allowed to deviate more (whilst still being treated as equivalent) then their contribution to the error reduction will be less. Note though, that even with a high normalised error tolerance of 0.25, the operator still stably reduces the mean error across all buildings by more than a meter. This can be largely attributed to the correction of newly erected buildings (constructions) and demolished buildings (removals) - since each singleton case will implicitly contribute more to the geometric variance between the datasets than an alternate class of change with objects of similar scope (size/span). Further the graph also illustrates that at a normalised pairwise error tolerance of 0.0, the operator replaces all geometry resulting in building models with a near perfect ($<$0.01m) correspondence to the LiDAR. Such an update is tantamount (equivalent) to pure vectorisation of the LiDAR without reference to the CAD. At normalised pairwise error tolerances of greater than 0.25 the operator returns the minimal number of replacements as necessary to improve the CAD's accuracy however it is still bound by the requirement to ensure that the mean deviant distance is less than d_{max} (2m in these experiments), ensuring an error reducing response.

The graph below shows the individual contribution of each geometric operator to the global reduction in geometric error. In order to calculate an individual change operation's contribution to error, at each iteration only the results for objects edited by the operation are considered.

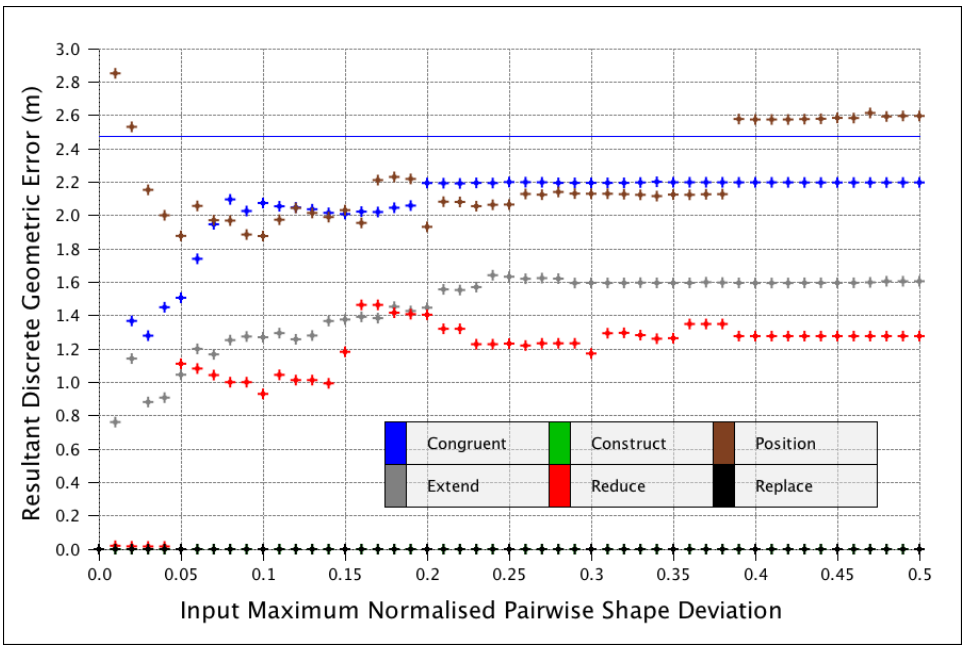

Figure 2.27: *graph of geometric error in meters (y-axis) against max normalised continuous pairwise error (x-axis) for individual update operators*

The blue line represents the original error between the CAD model and LiDAR points (prior to execution). This graph shows the dominance of the construct and replace operations in terms of raw accuracy improvement. Further (somewhat surprisingly) it demonstrates that the extend and reduce operations far out-performed the position operation in terms of their typical enhancement of deviant objects. Indeed the position response is temperamental which points towards a limitation of the discretised representations.

The results indicate that the reduction in geometric error is bound by the discretisation step - which is constrained by the resolution of the LiDAR. In particular considering the contribution of the individual operators to the building-to-building error reduction suggests that the only means to guarantee near-zero error is to base the updated model solely on LiDAR data. Ultimately the proportion of the original CAD model that is retained in the updated model is controlled by the maximum normalised pairwise deviation. For strict tolerances (such as less than 0.02) the operator is equivalent to a dense interpolation of the LiDAR points. As the tolerance is relaxed more and more of the CAD model is deemed acceptable and thus included in the updated model. Whilst this acts as an intuitive way to control the level of acceptable deviation, the operator profiling points to the fact that the inclusion of original CAD objects contributes least to optimising the physical accuracy. Although computationally expensive, vectorising pointsets directly does the most work in terms of ensuring a scene is accurate. Note however that the quality (or visual detail) of the vectorised models is bound by the resolution of the LiDAR scans used. So whilst a strict pairwise deviance tolerance ensures a near-perfect correspondence to the discrete LiDAR points, it is not guaranteed to return models of equal detail to the original CAD.

Type Classifier Results

The results of applying the object-type classifier to the segmented object clusters are presented in the following table and precision-recall graph.

	Man-made	Organic	Unknown
Ground-Truth Count	1404	1098	160
Classifier Total Count	1655	1007	0
Recall Correct Count	1316/1404	871/1098	0/160
Recall Correct Rate	93.7%	79.3%	0%
True Positive Count	1316/1655	871/1007	0/0
False Positive Count	339/1655	136/1007	0/0
True Positive Rate	79.5%	86.4%	0%
False Positive Rate	20.5%	13.6%	0%

Figure 2.28: *test result - object type classifier*

The table above summarises the accuracy of the object type detector based on a single execution - relative to the ground truth labelled object-types. The graph below illustrates the precision and recall of this classifier. Note that the unknown (or ambiguous class) is also included for completeness.

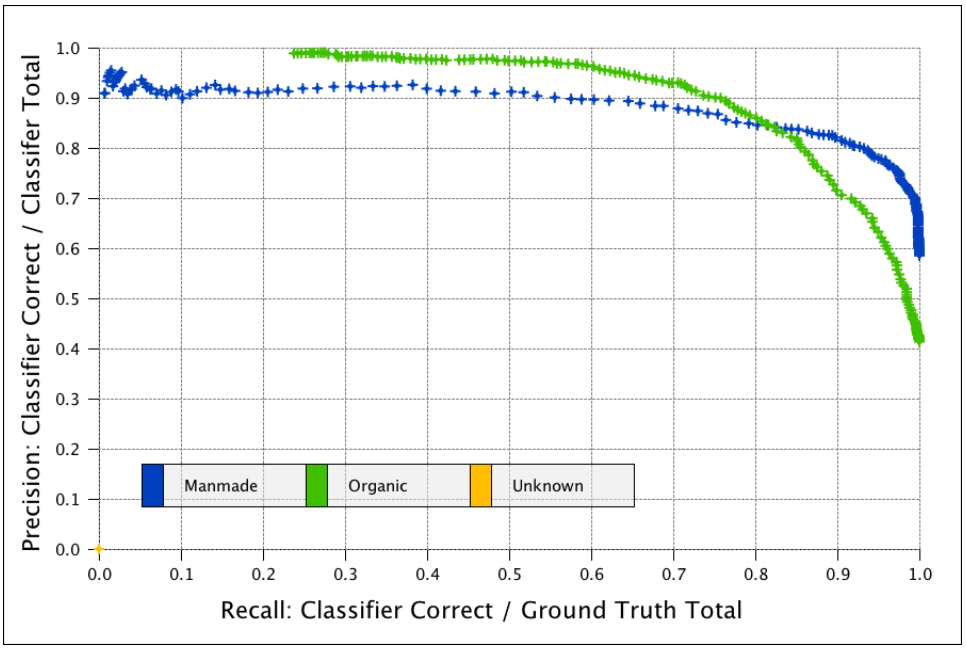

Figure 2.29: *precision recall graph for the individual object type classifier: man-made, organic, unknown.*

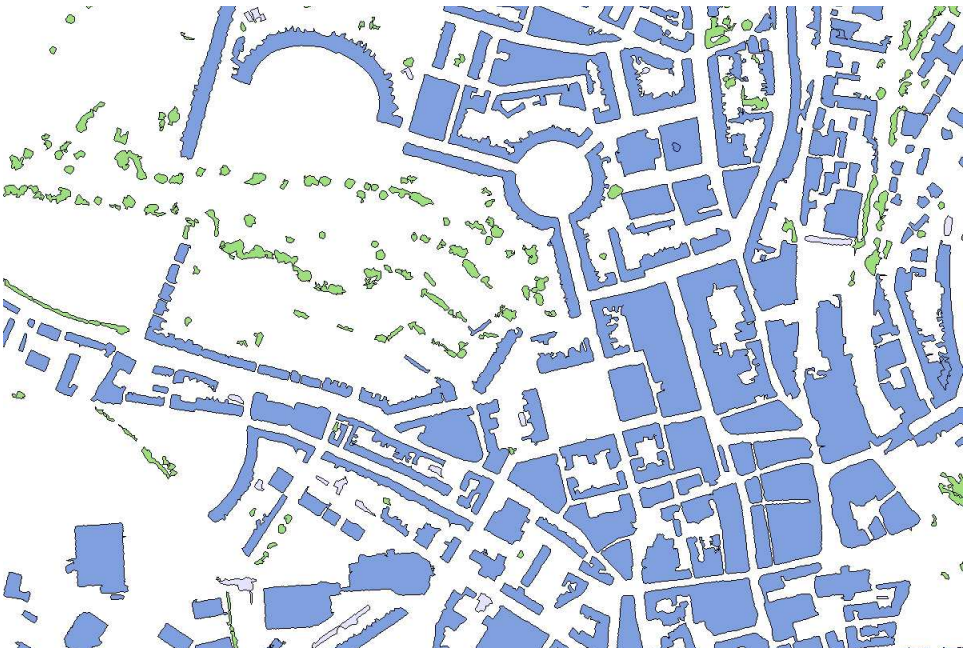

Figure 2.30: *results of type-classifier: with objects coloured by type for the 1m Bath dataset (blue: manmade, green: organic).*

The result of the object type classifier points to a promising architectural object filter, but the requirement for further refinement. At its current performance it achieves 93.7% building object recall rate across the human labelled objects in the City of Bath, with a detection precision of 79.5%. This means for 1000 buildings, approximately 937, would be correctly detected however the classifier would also return up to 200 odd additional objects (that are not buildings) that would need discarding. Though it may be easier for a human operator to discard rather than recall objects, at the current performance the object type classifier would be insufficient for a fully automated update operator that omitted a post-update validation stage. Figure 2.29 also indicates the type-classifiers sweet-spot as lying at around 85% detection precision with a corresponding recall rate of 85%. It shows that to achieve a building recall rate of 95% and upwards the detection precision would fall to below 70% which would mean clutter objects and vegetation would be more likely to slip through the filtering stage.

Variance Classifier Results

The results of applying the variance classifier to the pairwise object matches are presented in the following table. This test aimed to assess the stability (distinguishability) of the full set of deviance types defined by our scheme.

Remember that this is based upon comparing the classifier's response to a human's desire and not an explicit binary (true/false) distinction. Unlike the type classifier, for which the entire dataset was labelled, this section deals with the response for a subset of the dataset. The subset used was selected based on the clarity of the alteration. This is to determine how the operator responds relative to pairwise variances that can be unambiguously classified by a human. In essence

only clearly defined object variances were considered. The aim of this to to see how close the formal logic comes to describing a human's perception of change.

	Congruent	Construct	Position	Extend	Reduce	Replace
Ground-Truth Count	54	542	20	14	11	30
Classifier Total Count	51	542	14	19	10	35
Recall Correct Count	46/54	542/542	12/20	13/14	10/11	27/30
Recall Correct Rate	85.2%	100.0%	60%	92.8%	90.9%	90.0%
True Positive Count	46/51	542/542	12/14	13/19	10/10	27/35
False Positive Count	5/51	0/542	2/14	6/19	0/10	8/35
True Positive Rate	90.2%	100.0%	85.7%	68.4%	100.0%	77.1%
False Positive Rate	9.8%	0.0%	14.3%	31.6%	0.0%	22.9%

Figure 2.31: *test result for the update operation classifier*

In keeping with the previous format, the preceding table demonstrates the response for a single iteration, whilst the graph following illustrates the precision and recall as the normalised and discrete error tolerances were varied.

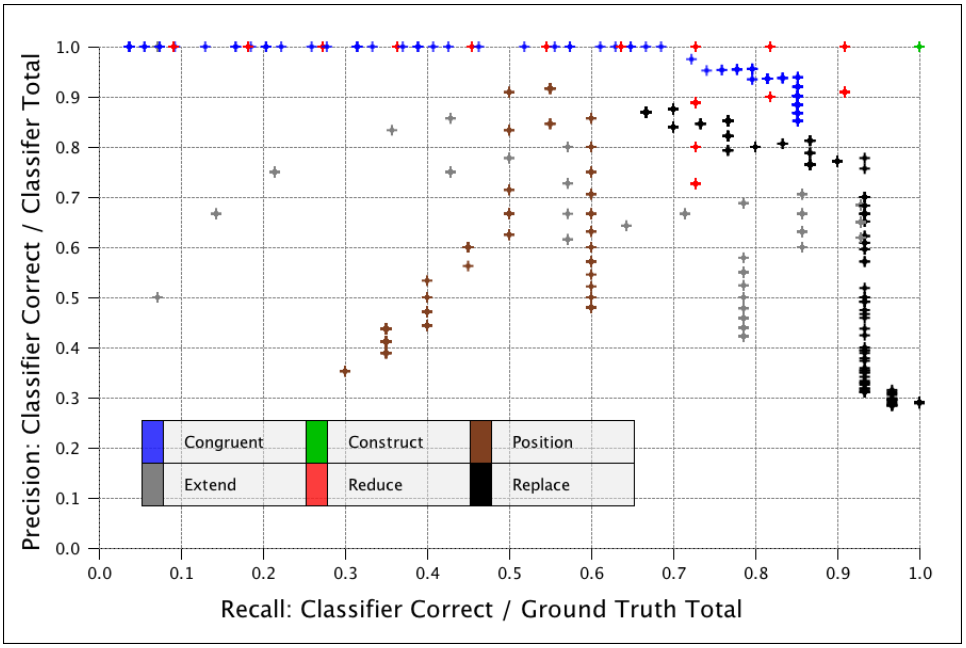

Figure 2.32: *precision recall graph for the pairwise variance classifier*

From this it is clear that the position deformation is the hardest for the pairwise deviance classifier to stably detect. Somewhat surprisingly the extend and reduce deformations were consistently well-behaved, despite the potential ambiguity with which a human may label each when each could equivalently be replaced. The overall performance of the pairwise deformation classifier indicates the feasibility of detecting multi-modal architectural change at city-scale using a formal geometric logic. The key feature is that it is based solely on the product of geometric

evaluation and is not biased by potential training strategies. The only concern is that a human's desire for a specific operation may not map to the class of deformation actually represented by the object. This is a challenge and whilst machine learning provides a plethora of methods for framing classifiers based on a set of examples, the prevalent problem is that the resultant classifier could potentially over-fit the training data. The advantage of the closed form approach is that the operator is independent of the dataset used during development.

2.5 Analysis and Evaluation

Having profiled the resulting performance of the semantic change detector - analysis of its behaviour follows, in order to draw insights from the approach taken. This section evaluates the operator both theoretically and in terms of the physical implementation. It provides a high-level examination of the operator that clarifies its strengths, limitations and potential improvements.

2.5.1 Type Classification

The results of the object type classifier points to a promising architectural object filter, but the requirement for further refinement. At its current performance it achieves 93.7% building object recall rate across the human labelled objects in the City of Bath, with a detection precision of 79.5%.

It is important to note that (as discussed in the previous sections) the function used to identify buildings could just as easily be replaced with an alternative method of performing linear discriminant analysis. In this respect the type-classification is not reliant on the formulation presented. It is simply a filtering stage and could be handled with a support vector machine or neural network (as examples). The critical aspect of the stage (from an engineering perspective) is maximising the recall and precision without the loss of generality. In essence the type classifier's decision logic should be generic enough to effectively handle different types of urban environment (town, inner-city, industrial, residential...), scanned at arbitrary resolutions and with (potentially) non-uniform sensing noise. The requirement to be able to generalise, coupled with the open-nature of this problem means that it is unlikely a classifier can achieve 100% recall and precision - however anything above 90-95% can be considered effective - since such performance will handle the majority of the task without sacrificing the ability to generalise. This represents a trade-off made in order to prevent the classifier being overly learnt on the data used during development - particularly because the aim is to be able to generalise to different classes of aerial laser scan, beyond simply structured DEMs. Further by considering the requirement for tractability in the detector the utility of the heuristic function proposed is evident. Unlike fuzzy-logic, a Bayesian formation or a decision tree, the interaction of each term in the linear-classifier maps to a high-level semantic directive that is not bound to a single training dataset.

Abstractly the type-classifier enforces the following observations:

- greater object volume or surface-area implies a building
- the dominance of planarity across an object implies a building
- the dominance of continuity across an object implies a building

- disparate clusters of points imply an organic or ambiguous object

The balance of these high-level rules are controlled by the input type-classifier weights - a four component vector. Sensible weights were arrived at empirically - and as such the performance could be improved by adaptively learning the weights using an unsupervised algorithm. Although earlier tests tried this (in order to enhance the precision-recall balance from the mid-80s to mid-90s %) the outcome was that the level of generalisation dropped significantly and the weights learnt (although performing better on the training dataset) failed to generalise and produced inferior results when applied to validation sets (with different scan-resolutions). This represents a compromise made to enhance the robustness of the type-classifier. For small-to-mid sized datasets (1-10 million points), the function handles the majority of the filtering leaving between roughly 5-15% of the segmented object descriptors to be manually verified by a human-operator. For smaller datasets this can be considered reasonable since the time required to manually filter a fraction of a number of buildings (tens to thousands) is small (typically minutes) relative to setup, training and validation of an unsupervised learning algorithm. However in the case of large city-scale datasets (containing tens or hundreds of thousands of buildings) the requirement for manual intervention must be limited - and as such the use of a unsupervised-weight refinement technique is appropriate. Whilst this breaks the generality of the classifier it is not unreasonable to store a set of distinct weight vectors inline for each geographic region or locale processed by an architectural or surveying firm.

2.5.2 Variance Classification

The performance of the variance classification points towards an effective means to identify actionable differences between out-dated CAD models and up-to-date LiDAR scans. The response of the construct and remove classes is practically infallible. The extend, reduce and replace operators respond well (with precision recall in the 90+% area. However the congruent and position classes seem particularly sensitive to noise (at roughly 85% and 60% precision-recall). Determining the right discrete and normalised error bounds can be an iterative process. However once determined they generalise well to different datasets at different resolutions. The key factor in the limited response of the position class is the method of determining the aligning transform. Since we rely on a one-to-one mapping between CAD elevation points and aerial LiDAR points (for efficient differencing), the method of registering two pairwise building clusters does not exploit the sub-pixel registration routine and as such limits the precision that can be attained. This problem is compounded by the presence of sensing noise which introduces undetermined deviance between the two representations. The underlying problem is that in the presence of noise, geometries that are visually semantically equivalent to a human, may deviate so as to fail an error-driven equivalency test irrespective of their alignment. To combat this you could increase the point-to-point tolerance to a large number of meters (10+) and reduce the minimum required normalised intersection over union to a small number (such as 0.05 - $1/20^{th}$ of the pair must overlap), however this invariably alters the behaviour of the other mutually exclusive classes and generally degrades performance. In this sense changing the tolerances does not address the problem but rather incorrectly causes other classes of deviance to converge on the position operation. During development we tried ICP and the sub-pixel sliding window registration (using the Euclidean distance metric

to combat the invalidation of the one-to-one mapping between the registrees), but observed little change to the response of the position-class whilst incurring much larger runtimes that exceeded that of the segmentation stage). Unfortunately we failed to arrive at a simple solution to ensuring each and every position-able building would not be treated as a replacement, that did not negatively impact the overall performance of the operator. One thing we noticed was that although the normalised measure worked well for smaller buildings for larger buildings it tended to treat roughly equivalent buildings as deviant. This was primarily a result of features present in the LiDAR that may not have been modelled explicitly. Such as points corresponding to architectural adornments and temporal objects on roofs. In particular we realised that there was a non-linear relationship between our perceptual tolerance of error and the extents (size, volume, surface-area) of the geometric entity - which we failed to characterise in the semantic change detector. In hindsight taking the scale of an object into-account would provide a means to adaptively alter the equivalency tolerances so as to ensure scale-appropriate behaviour - i.e. it would be more forgiving of small deviant features on larger objects.

The key challenge with such an approach however is the derivation of the means of varying the weights. At a high level, the desire is quite intuitive - i.e. the operator should relax the discrete error tolerance as the size of a building increases. The critical problem with this is the introduction of variable error characteristics across individual buildings. Whereas at present a uniform normalised and discrete error is used for all buildings, such a strategy would treat buildings of different scales as having different quantitative requirements for each variance class. As such one would no longer be able to say that the class labels represent the operations required to correct the entire scene to within a uniform error-tolerance, but rather to correct each building to within a unique (somewhat arbitrary) tolerance that is more relaxed for large buildings and stricter for smaller buildings. Though this would provide greater benefit the logic becomes convoluted because in order to determine how to alter error-tolerances in line with scale changes, manually-constructed human labelled data would need to be analysed ahead of time (offline) or at runtime. This breaks the independence of the change-detector and would mean the logic becomes quantitively tied to the domain. This trade-off represents a decision made in the favour the generality of the technique. However it stands as a key area for further investigation.

Another key limitation of this approach is the manner of resolving multiplicities. The operator identifies multiplicitous objects and treats them as replacements. At an object level this is not a problem. However at the component level it would sometimes be desirable to be able to chain a set of classes rather than yield a replacement. For example a building may exhibit a reduction coupled with two extensions. In such an instance, if the scope of the deviant regions is small relative to the congruent regions then a CAD-technician may prefer a sequence of extends and reduces rather than a single replace - so as to maximise the inclusion of the original CAD model.

This is not only a limitation in the case of multiplicitous mappings, but also in the case of multiple alterations applied to an object. To address this the obvious solution is to cluster deviant regions at the level of individual building components and yield a sequence of variance classes each corresponding to a component-level change. However this introduces the requirement to further segment each deviance which (though trivial for simple cases) quickly becomes ill-defined in more

complex instances such as curvature and non-planarity. Hence the decision was taken to omit multiple-level classification, in order to conform to the behavioural requirement for tractability and efficiency. Although the simplified strategy works well, this particular limitation stands as a clear area for further investigation.

2.5.3 Computational Efficiency

The computational efficiency of each of the components implemented in this work is summarised in the following table. It contains the mean run-times (in seconds) for processing varying sized subsets of the City of Bath dataset over a series of ten executions. The operator was profiled on a quad-core i7, with 16GB of RAM, using multi-threading. The column headings denote the square-area of the test region and resolution used (the sample step).

Process	$1km^2$ (2m)	$1km^2$ (1m)	$5km^2$ (1m)
Registration	1.24	3.85	9.77
Variance-Detection	0.56	1.73	6.04
Segmentation	3.04	12.39	72.25
Classification	0.39	1.20	4.23
Visualise-Document	19.55	86.21	352.94
Total-Runtime	24.78	105.38	445.23

Figure 2.33: *Average process run-times (in seconds) for a series of 10 executions of the complete semantic-change-detector*

The results indicate a roughly linear growth in run-time relative to the size of the input discrete point-sets. For the largest datasets used ($5km^2$) the entire multi-modal semantic change detection process takes just under 8 minutes - with the bulk of the time being spent in the visualisation and documentation stage. The operator handles large models surprisingly quickly - however it returns dense models for visualisation. Further research could focus on methods of decimating the models without the loss of accuracy and without incurring prohibitive run-times. By exploiting lower dimension representations the variancing is almost instantaneous. However in considering the results in synthetic dataset experiments (particularly between 3D geometric pairs) it became clear that the efficiency of the variancing is a product of the Chebyshev distance measure. When the operator is applied to unstructured data (with the Euclidean distance metric), the associated execution time grows exponentially inline with the radius used to compute each point's neighbourhood and the density of the input point-sets. However with the structured parallax geometries, computing point-neighbourhoods is not required (since they are implicit in the representation), and this enables the differencing of CAD and LiDAR to run in roughly linear-time.

2.5.4 Qualitative Evaluation

This final portion of the evaluation section considers the high-level qualitative strengths and limitations of the semantic change detector.

Strengths

Efficiently Identifies Object-Level Modelling Errors - the operator quickly labels all significant sources of geometric error between out-dated CAD and new-acquired LiDAR in an abstract yet robust manner. Unlike the pre-existing strategies to change detection, this operator considers variance at the level of objects as opposed to individual points and thus classifies meaningful (actionable, semantically rich) descriptions of temporal alteration between airborne architectural datasets.

Enables Selective Reconstruction - culling the reconstructive process based on whether or not a building has actually changed since it was last modelled.

Generalised Boolean Logic applicable in 2D and 3D - the variance classifier operates equivalently on discrete and continuous pairwise geometries.

Limitations

Low-Resolution Laser-Scan Data vs High Resolution CAD Data - one of the key problems with the experimental tests on the city of Bath dataset, is the limited resolution of the input DSM points. This has a significant impact on the accuracy of the analysis. Although it demonstrates that the operator works well even with low resolution (1m point-spacing) point-sets, in a ideal world higher resolution point-data would have also been used (to test the limits of the change-detector at higher levels of detail). This is a vital issue with the use of off-the-shelf scan datasets. In particular you'll notice that in the qualitative figures the level of detail present in the original CAD model isn't preserved in the labelled deviant regions - each region acts merely as a positional indicator and fails to capture subtle roof-details in visualisation.

Uniform Discrete and Continuous Error Tolerances - results in the operator treating all buildings with fixed error tolerances, irrespective of their size. As discussed, this could be viewed as undesirable. The challenging aspect of addressing this lies in determining a generic method of varying the error tolerance in line with the spatial scale of each pair of overlapping buildings.

Rotational Invariance - is not explicitly supported by the variance classifier. At present the expectation is that the transformation model used during registration is only responsible for determining a translational offset. By exploiting an alternative registration method (that estimates orientation), the utility of the operator could be further enhanced. The future investigative track in this regard is the implementation and evaluation of deterministic rigid-body registration functions designed to operate efficiently on 2.5D (parallax) elevation models. Note: given the nature of the data, the inclusion of rotational-invariance can be considered an optional feature due to the expectation of *north-coherency* between the CAD and LiDAR.

Dense Boundary Visualisation Strategies - are desirable for the efficiency with which they can be constructed from sets of sampled points, however they lack the topological clarity associated with manually constructed models - making the deviant regions inappropriate for direct use in a fully automatic temporal update modelling function. Whilst they serve their purpose as a means to denote the critical geometric errors - they do not yet provide a suitable representation for further updating the geometry because the quality of the building representations present in the updated CAD would deteriorate - and subtle details would be lost.

Limited Testing Data restricted the scope of the evaluation on real-world datasets to the city of Bath. The problem is access to out-dated CAD models and newly-acquired laser-scans. Thankfully the department of Architecture and Civil Engineering at the University of Bath were able to provide access to the city of Bath model, and the Environment Agency the airborne range scan of the city. Whilst this chapter demonstrates the validity of the idea, it failed to provide more rigorous analysis using different cities scanned and modelled at different resolutions and levels of detail (respectively). This is an on going problem that is being addressed.

2.6 Discussion and Summary

In conclusion, this chapter presented a simple yet effective semantic change detector - designed to provide actionable descriptions of architectural variance between old CAD models and new point-clouds. It delved into how we arrived at the 6 classes of deviance detected and analysed the performance of the operator with synthesised data and on the City of Bath dataset.

Whilst the results are promising they naturally lead to the realisation that without robust automatic modelling strategies the utility of this operator is limited. Essentially in order to bring automatic temporal updates into being it is clear that reliance on dense reconstructions is insufficient. Though we now have a stable means of quickly identifying and classifying geometric errors, we still lack effective methods of dealing with the requirements to construct, extend and replace building-geometry. Though it is feasible to use dense surfaces for still rendering, for interactive scenes they become prohibitive to maintaining a steady framerate. We conclude this chapter noting that although the overall idea is sound, the critical bottleneck to effective application, lies in the ability to take in arbitrary point-clusters and return clean and accurate, sparse boundary-representation models.

For reference, the key concepts covered in this chapter are revised below:

- **Sliding-Window Sub-Pixel Registration** as a deterministic method of aligning the out-dated CAD model and up-to-date LiDAR scan
- **Difference-of-Elevation Models** as a means of slicing a terrain-normalised digital elevation model into two models - salient and terrain
- **Scan-Conversion Segmentation** in order to cluster disjoint connected components (neighbouring sets of points) in linear time
- **Linear Type-Classifier + Ambiguity Check** to filter vegetation, vehicles, street-furniture and similar clutter from subsequent analysis
- **Pairwise Analysis** as a method of abstractly characterising the difference between two rigid geometric representations - boolean operations implement the predicates and constructions: PairMatch, CongruentDeviantRatio, PostiveMassRatio, NegativeMassRatio, MonoModalMeasure, DiscreteEquivalencyTest, ContinuousEquivalencyTest
- **Resolving Multiplicities + Handling Sensing Noise** to deal with the two edge-cases (a one-to-many or many-to-many mapping) in pairwise analysis and ensure robust analysis in the presence of undesirable artefacts through the use of continuous and discrete measures

- **Variance Classes** - semantically-rich descriptions of geometric change : Congruent, Construct, Remove, Position, Extend, Reduce, Replace
- **Visualisation + Documentation** of the variance present between CAD and LiDAR to support manual geometric updates by CAD technicians

Finally the complete semantic change detector defined in this chapter is restated. The classifier acts as the temporal analysis module of the urban update mechanism sought by this research. It enables selective-reconstruction.

This represents the key contribution to knowledge presented in this chapter - it is a closed-form semantic classifier for determining the differences between arbitrary geometric representations - as a product of boolean predicates. This generalised logic applies equivalently to both two dimensional and three dimensional geometric comparisons and is applicable to both continuous object representations and discretized object representations.

Multi-Modal Architectural Deviance Detector

$$classify\_update\_operation\_with\_noise(A, B, t, d_{max}, e_{max}) = \tag{2.22}$$

$$\begin{cases} construct, & \text{if } (A = \varnothing)\ \&\ (B \neq \varnothing) \\ remove, & \text{if } (A \neq \varnothing)\ \&\ (B = \varnothing) \\ position, & \text{if } (A \neq \varnothing)\ \&\ (B \neq \varnothing)\ \&\ (mean_{err}(A \times t, B) < d_{max}) \\ extend, & \text{if } \left(\frac{\int (A - B)}{\int (A \oplus B} < e_{max}\right)\ \&\ \left(\frac{\int (B \oplus A)}{\int (A \cap B)} < 1\right) \\ reduce, & \text{if } \left(\frac{\int (B - A)}{\int (A \oplus B} < e_{max}\right)\ \&\ \left(\frac{\int (B \oplus A)}{\int (A \cap B)} < 1\right) \\ replace, & \text{if } \left(\frac{\int (A \oplus B)}{\int (A \cap B)} \geq 1\right)\ || \\ & \left(\frac{max(\int (A - B), \int (B - A))}{\int (A \cap B)} \geq e_{max}\right) \end{cases}$$

where the input arguments A, B, t, d_{max} and e_{max} correspond to:

$A \rightarrow$ the out-dated (old) geometric representation
$B \rightarrow$ the up-to-date (newly-acquired) geometric representation
$t \rightarrow$ the error minimising aligning transformation
$d_{max} \rightarrow$ the discrete (point-to-point) maximum deviant distance
$e_{max} \rightarrow$ the normalised (intersect-over-union) maximum non-overlap ratio

Chapter 3

Aerial Mass Reconstruction

What is it?

A data-driven algorithm for direct vectorization of aerial LiDAR and an extension to support efficient parametric primitive based model optimisation.

Why does it exist?

In order to enable robust, efficient, accurate, unconstrained sparse reconstruction of top-down architectural models from airborne lasers-scans.

How does it work?

By efficiently segmenting the range-images using the difference-of-elevations and area-maximising region detection - followed by sparse vectorization and graph-based constrained shape refinement and non-linear optimisation.

parallax building blocks...

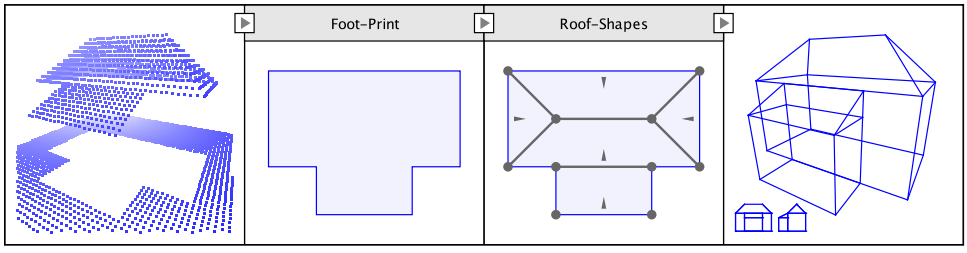

Figure 3.1: *an overview of the key stages in aerial mass reconstruction - from left to right: an input aerial point-cloud, detected 2D footprint, segmented and refined 2.5D roofshapes, the resultant projection model.*

3.1 Overview

This chapter addresses the problem of efficiently reconstructing sparse building models from airborne point-clouds. It introduces an advanced variant of the segmentation technique discussed in the previous chapter, which seeks to address the limitations of using dense-surface representations to characterise newly acquired building geometry. This aim of this chapter is to define and analyse reconstructive operators capable of yielding sparse (compact), semantically-rich massing-models given parallax (2.5D) LiDAR scans. The context of the chapter is in 3D mapping and surveying with a particular focus on the suitability of generated geometry for architectural visualisation.

The techniques presented build upon those covered in the previous chapter - however now the focus is geometry recovery as opposed to analysis.

An additional key desire is to recover geometric models that are analogous to models constructed manually by a human CAD technician. In order to enable this a useful shape approximation paradigm is employed. The *maximal-area : minimal-primitives* principle. At a high level the principle aims to ensure that in creating sparse building mass models, the operator uses as few geometric primitives as required to meet a user supplied error tolerance. This a vastly different from the pre-existing techniques that frame the reconstructive process as a product of optimising against high-level imposed topological features - such as regularity [167] or smoothness [107].

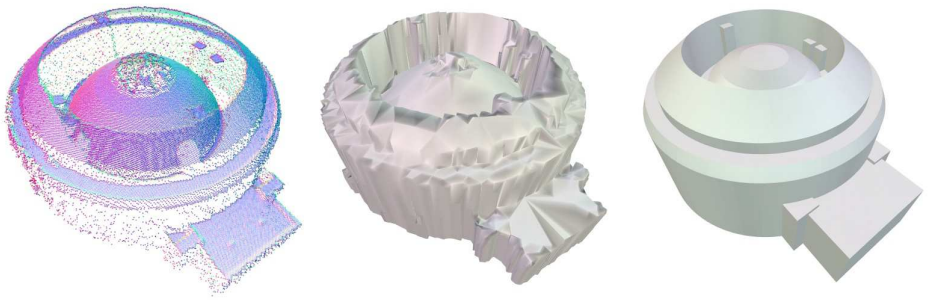

Figure 3.2: *a real world example to help clarify the maximal-area, minimum-primitives principle - (left) an aerial scan of the Manchester public library, (center) the result of automatic reconstruction with 2.5 dual-contouring [165] and (right) a manually constructed CAD model - this figure illustrates the significance of the difference in the visual quality of geometric representations resulting from automatic techniques relative to manual human creation - the MAMP principle acts as a means of formally describing this difference and guiding algorithms towards models similar to the manually created geometry - in-particular note how the libraries' quadrilateral entrance is represented as a single object in the manual model whilst the automatic model uses a multitude of components which results in noticeable shading artefacts - note additionally that this figure also illustrates a key limitation of most data-driven strategies - i.e. their ability to handle (effectively represent) curvature.*

The maximal-area : minimal-primitives (MAMP) principle is an incredibly simple yet highly effective paradigm for airborne building reconstruction, and as such a significant amount of attention is given to its exposition.

In this chapter the computational performance of the algorithm presented is evaluated in terms of the following three critical attributes.

- Geometric Model Accuracy
- Execution Time - Algorithm Runtime
- Brevity - Level-of-Compression - Compactness

To be clear, geometric model accuracy refers to how similar a reconstructed model is to its source scan points - i.e. the geometric error between input and output, execution time deals with the growth in algorithm runtime as a product of the size of the input (i.e. its scalability), and model brevity is based on the ratio of the number of vertices in an output model to the number of points in an input scan.

Recall from the literature review and background that these are the performance properties that are ubiquitously shared by all building reconstruction methods (and indeed all shape approximation methods).

To help clarify examples of concrete use-cases for the airborne reconstruction algorithm defined in this chapter - one could conceive exploiting it:

1. As a method of automatically creating accurate 2D building footprints and roof-shapes for digital maps of large urban districts.
2. As a means of supporting the process of automatically analysing the types of architectural geometry present in a city-scale urban region.
3. As an automatic reconstruction operator for recovering aerial massing models of planar and non-planar architecture geometry from LiDAR - in order to enable data-driven 3D simulation of the physical world.

For reference the key objectives, requirements and behavioural desires of the operator are stated following. Remember this is the method of turning aerial point-clouds into sparse parallax building mass-models.

- Geometric Accuracy (correctness/precise/error-bounded results) - such that the reconstructed geometry can be used for surveying tasks, for which it is imperative that the magnitude of each model's error is quantified (and minimal). The volume of each reconstructed mass should mirror the volume of the building's mass in the physical scene.
- Maximising of the mean polygonal face surface area - such that the reconstructed geometry embodies the topological structure typically associated with manually constructed models. This is equivalent to saying the operator should use the fewest (as few-as-is-possible-given-the-nature-of-the-input-laser-scan) geometric primitives required to model each point-set whilst conforming to a user supplied error tolerance.
- Computationally Efficient - in the sense it exhibits roughly linear growth in runtime as a product of the number of input-points, and not the exponential decay in execution time that is commonly associated with the prevalent sparse reconstruction operators.
- Robust to Geometric Degeneracy and Sensing Noise - such that anomalies (like ghost/shadow points, non-uniform sampling, and low and high frequency scanner noise) do not prevent the recovery of accurate models - or detrimentally influence the reconstructed geometry.

- Independent of Point-Spacing (Density/Resolution) - in order to ensure sparse architectural masses can be recovered for different (/irrespective of the) types (classes) of airborne laser-scan supplied as input.
- Independent of Constraining Priors - such as the manhattan assumption or the enforcement of pre-determined principal angles - in order to ensure true data-driven polygonisation, that is capable of modelling virtually any class of building without limitations on geometric form.
- Suitable for structured and un-structured airborne range-data - in order to enable its ubiquitous exploitation on both scans from fixed-wing aircraft and scans from unmanned drones such as quadracopters.
- Deterministic (Non-Stochastic) - such that the operator is capable of producing the same (an identical) result, given the same inputs repeatedly (time and time again). In essence the output geometry should be a direct product of the input laser-scans and the control parameters - nothing more. Reliance on random-sampling strategies (such as RANSAC) should be completely omitted in order to enhance stability.
- Controllable via Continuous Scale-Space Level-of-Detail - such that an end-user of the operator can intuitively vary the scale of features present in the reconstructed geometry - in order to include or omit smaller elements like chimneys and air-conditioning units.
- Capable of Implicitly and Explicitly Modelling Curvature - in the sense that it can automatically characterise non-linear roof-surfaces using both data-driven and model-based reconstructive paradigms.
- Capable of Extension via User-Plugins - such that its reconstructive modelling capabilities can be added to, post-compilation. In essence, the desire is to be able to plug-in new geometry detectors later down the line, without having to alter the operators underlying architecture.

Whilst some of the properties are obvious (for example computational efficiency, geometric accuracy and robustness to sensing noise), others are less apparent and largely the result of insights drawn from considering the limitations of the pre-existing approaches. For example the requirement to maximise the mean polygon face surface area and the independence from constraining shape-priors are key to addressing the bloat associated with the reconstructive process and the limitations of using fixed-form heuristic shape-approximation methods. The chapter shall refer back to these properties during the discussion of the key algorithmic stages. The vital thing to remain aware of throughout, is the result of demanding these behaviour characteristics of the technique proposed. There are critical benefits and advantages that derive from conforming to these properties. Ultimately the reason for demanding these behaviour characteristics is to ensure the reconstructive method is capable of fast, accurate and sparse top-down building model recovery.

The remainder of this chapter is structured as follows:

- The background and context section recaps the key pre-existing approaches to airborne building reconstruction from laser-scanned point-clouds. This includes considering the strengths and limitations of these dominant methods.

The aim is to elucidate the research that inspired and informed the development of the proposed automatic mass-modelling operator MAMMAL.

- The methodology section presents the approach taken to automatically reconstructing sparse building models from aerial LiDAR. It begins with a complete overview of the algorithm and then progresses to each of the processing stages: segmentation, vectorisation, projection and optimisation.
- The experimental results section enumerates the outcomes of profiling the performance and behaviour of MAMMAL using multiple aerial scans of the cities of Bath, London and Manchester (UK) (at various point-spacings). The section presents quantitative and qualitative measures of the algorithms competence. A handful of smaller synthetic datasets are also considered to help clarify abstractly where the approach fails in controlled experiments.
- The analysis and evaluation section contains high-level analysis of the airborne reconstruction operator. It considers and expands upon the results in order to draw deeper insight from the experiments. Although the primary focus is quantitative analysis (determining the geometric-accuracy, geometric-quality and level-of-compression of the reconstructive routines), it also outlines the key qualitative strengths and limitations of the proposed approach.
- The newly-added improvements and enhancements section details the potential alterations that stand as future research to develop the operator further. Each of the investigative tracks discussed aims to address one of the three key evaluative factors - efficiency, accuracy or structural quality.
- The discussions and summary section provides a synopsis of this chapter - revising the aims and outcomes and commenting on the implications moving forward. It seeks to be concise and uses bullet-points to reiterate the key-points, ideas and concepts introduced in this chapter.

Note: one of the key differences between this chapter and the previous chapter is the manner of evaluation. In the case of semantic change detection the efficacy of the operator can only be determined as a product of human input (desired) variance labels. However in the case of aerial mass reconstruction the key performance properties are purely a product of the variance between the input and output. Essentially evaluation is simpler because it only requires the characteristics of the output models and their deviance to the input point-cloud be evaluated. This is vitally important - since it both greatly simplifies performance analysis (by removing the need for human input) and (as a result) ensures that evaluation (in particular quantifying algorithmic error) can be fully automated. This *win* represents one of the key differences between classification problems and approximation problems.

Note: through-out this chapter, various buildings from the City of Bath, London and Manchester datasets are used to illustrate the key-principles and stages. In each instance the dataset is cited but as a high-level synopsis simply be aware that the Bath dataset is the same airborne range-scan used in the previous chapter's semantic change detector - sampled at 1m point-spacing. The London dataset covers a 10x10km^2 region of the UK capitol - sampled at 50cm point spacing (and represents an area of interest to one of the parties that funded this research). Whilst the Manchester dataset is composed of scans at 25cm point-spacing. The Bath dataset (although low-resolution) is useful because of the corresponding CAD

model which enabled qualitative comparisons to be made easily. The London dataset is useful because it covers a reasonably large area that is also extremely indicative of the architecture present in highly populated urban centers (i.e. it represents a vital real-world example of a city suitable for temporal revisions - that happens to also be of industrial interest). The Manchester dataset is useful because it represents the higher resolution end of the currently available off-the-shelf (open-access) data : and as such enables evaluation of the algorithms behaviour as the point-spacing varies. It is also another good example of a city suitable for temporal revision (since it changes frequently). Additionally the Manchester dataset contains a number of highly irregular buildings that help determine how well generalised and applicable to real architecture the algorithm is. Although higher resolution airborne scans can be achieved (50ppm+) such datasets are often commisioned. In essence each dataset embodies key characteristics that enhance the evaluation and analysis of the algorithm and its physical performance. Together they help ensure the techniques proposed adapt well to different city-scale laser-scans, and are not simply optimised for a specific geographic region.

3.2 Background and Context

3.2.1 Key Related Work

This section briefly recaps the most relevant related work.

It provides a synopsis of the most relevant pieces of research that have inspired and informed the approach to aerial reconstruction proposed in this chapter. Although (in the strictest sense) the newly proposed method is actually nothing like them (in terms of its operation and performance traits), these two contributions represent the current leading strategies. One is data-driven and one is stochastic.

Note: that due to the fact that both have already been examined in the case-studies presented in the literature review - the aim here is simply to ensure the key aspects of each are fresh in the readers mind.

Zhou & Neumann (2008-2013)

Qianyi Zhou's and Ulrich Neumann's 2.5D Dual Contouring collection [167], [164], [165], [163], [166] - represents the dominant data-driven strategy that is referred to in this chapter and the wider academic community. It has developed greatly over time - yet has reached a point of stability and prevails in industrial contexts. From an intuitive plane based mass-modelling method - it has gradually grown and been extended to include heuristic data-driven logics for enhancing model quality and dealing with irregularity. Note that although these extensions have increased its execution time (from seconds to minutes) is it still far more efficient (and direct a method) than Florent's and Mallet's approach. The key issue is that at low resolutions model quality can suffer relative to the hybrid method discussed next. Further more, their key topological enhancement (enforcing global regularities) is designed primarily to address roofs composed of planar pieces - which means its performance in the presence of curvature is not always satisfactory. However one thing in particular that sets Zhou's operators apart from other data-driven methods is the publicly accessible implementation - which makes performance analysis and evaluative comparisons possible. As a result, it is the most academically cited

approach within the field and acts as a basis for numerous other contributions in the domain. One thing in particular (that this author really admires about this approach) is the ideology underlying the proposed solution. Each sub-problem is addressed analytically rather than through trial and error. Zhou relies on insights regarding the built environment, alongside knowledge about the medium and frames the solution in terms of closed-form geometric predicates and constructions. For example he exploits Eigen-vector analysis during classification and a constraint based formulation to refine model topology. The result is a stable set of operators that can be relied upon to deliver robust results repeatedly.

Florent & Mallet (2008-Date)

Lafarge Florent's and Clement Mallet's generalised collection [71], [72], [73], [156], [67], [68] - represents the dominant hybrid strategy - and has also developed and advanced over time. However there has also been the gradual inclusion of more and stochastic techniques. On the one hand this is understandable (to an extent) - since their research group also investigates photogrammetric solutions (MVS) - alongside the reconstruction of actively sensed point-clouds. However this reliance on random sampling has had the negatory effect of dramatically increasing the execution time of their operators and they are currently orders of magnitude slower than Zhou's operators - taking hours to reconstruct city-scale scenes. Positively though this helps to mitigate sensing artefacts and ensures that higher-quality models can be produced even in the presence of lower resolution point clouds (at less that 1m point spacing for example). One vital aspect (that this author in-particular really admires about their approach) is that Florent, Mallet et. al were amongst the first to recognise the need for a conglomerate strategy that could freely mix distinct modelling paradigms in the construction of large urban scenes from airborne point-clouds. However somewhat detrimentally (to their own academic proliferation - depending upon one's viewpoint) they have patented aspects of their operators - which has resulted in far less formal evaluation of their strategy relative to Zhou's and Neumann's research.

***

In considering these archetypal contributions - one could say that Zhou and Neumann lean towards geometric operators, whilst Florent and Mallet tend towards computer vision strategies. This is the key distinction between them - and though both have their merits - ultimately the methodology proposed in this chapter aims more to follow in the footsteps of Zhou and Neumann. Note: that this does not mean that the MAMMAL algorithm replicates Zhou's approach - rather it points to the fact that from an ideological perspective MAMMAL also tends towards analytic geometric strategies in-place of traditional stochastic or probabilistic methods.

Before moving on to MAMMAL's methodology - it is important to reiterate that the novelty of MAMMAL is not a product of it solving a brand-new problem. Indeed automatic building modelling is one of the oldest applications of active-sensing. Rather the novelty is a product of the manner in which MAMMAL tackles a long-standing (yet challenging) problem - and (in particular) the computational performance of MAMMAL relative to these established methods. Essentially automatic top-down building modelling from point-clouds is not in and of itself a new phenomena. However fast, accurate AND sparse automatic building modelling from airborne point-clouds is both new and highly advantageous.

3.3 Methodology

The algorithm defined in this section is designed to robustly and efficiently extract clean, compact, parallax architectural massing models from airborne laser scans of urban regions - that are simultaneously accurate relative to the input and sparse enough to be rendered interactively at city-scale.

Figure 3.3: *Key Stages of the prescribed **MAMMAL** Algorithm → Maximal-Area 2.5D Mass Modelling of Airborne LiDAR*

Figure 3.3 provides a visual indication of the four key stages which involve:

1. Segmenting the input laser-scan to identify buildings and roof-shapes.
2. Vectorising the segmented roof-shapes to create 2D polygons.
3. Projecting the vectorised 2D polygons into 3D space by minimising the point to plane error - in order to generate parallax mass models.
4. Optimising the projected mass-models by non-linear procedurally-driven parametric shape refinement - based on an efficient error measure.

This part of the chapter discusses each of these stages, detailing for each the considerations and algorithmic steps undertaken. It begins with a complete overview of the airborne reconstruction operator, which is followed by the discussions of the segmentation, vectorisation, projection and optimisation.

3.3.1 Outline

This section provides a brief outline of the structure of the newly-proposed MAMMAL algorithm. It covers the input and output data and provides pseudo-code to clarify the lower-level behaviour details. For reference figures 3.4, 3.5 and 3.6 also illustrate the key concepts and stages employed.

Inputs-Outputs

The MAMMAL algorithm takes as input: a DSM .asc file - a digital surface model scan of the region to be reconstructed and a DTM .asc file - a corresponding digital terrain model of the same region - (which is generally derived from the DSM).

The algorithm also exploits the following control-parameters (supplied as input-arguments by an end-user): a scalar maximum point-to-edge footprint error (in meters), a scalar maximum mean point-to-plane roof-shape error (in meters), a scalar minimum intersection over union measure (a unit-less value in the range [0:1]), a scalar minimum building height (in meters), two scalars denoting the minimum and maximum area of building footprints (in meters^2), two scalars denoting the desired minimum and maximum surface area of building roof-shapes (in meters^2), a scalar maximum local variance to control segmentation (unit-less and positive), a flag specifying the model selection criteria to use for optimisation (a string corresponding to the name of an instance in the enumerated type *selector*),

a boolean denoting whether or not is should apply procedural model optimisation to the projected mass-models, and a boolean denoting whether or not watertight roof-shapes are required (which determines if the output models must be manifold or can be non-manifold).

MAMMAL returns the following outputs: a set of parallax 2.5D building mass-models (each composed of two sets of triangulated facets - the first set models each building's roof-shapes and the second, each building's walls), a set of approximate vegetation models (constructed by positioning simple tree icon models at the loci of the maximally inscribed discs of non-architectural segmented point-sets) and debugging data (in the form of colour-coded point-clouds for the segmentation stage, 2D master-plan shape files for the vectorisation stages and 3D error visualisation point-clouds and volumes for the projection and optimisation stages).

Note: the terms optimisation and parameterisation are used interchangeably in this chapter to refer to the process of procedural geometry enhancement.

Additionally (to aid exposition) figures 3.4 and 3.5 provide alternative pictographic representations of the inputs and outputs of the algorithm. Figure 3.4 details the inputs and outputs of the four key stages whilst figure 3.5 provides a flow-chart styled schematic of the complete method.

	Segmentation	Vectorisation	Projection	Parameterisation
Input	- Surface Points (DSM) - Terrain Points (DTM)	- Segmented Points	- Vector Shapes	- Object Descriptors - Projected Models
Algorithmic Stages	- Difference of Elevation Models - Vegatation Detection and Filtering - Roof-Component Detection - Noise-Cancellation	- Linear Boundary Edge Traversal - Vectorise Discrete Mask - Key-Edge Detection - Polygon Simplification - Water-Tighting	- Direct Offseting - Principal Component Analysis - Linear Projection - Non-Linear Projection	- Model Generation - Model Evaluation - Model Optimisation
Output	- Segmented Points	- Foot-Prints - Roof-Shapes	- Projected Models	- Parametric Models

Figure 3.4: *an outline of the key stages in aerial LiDAR vectorisation presented in tabular form - each column summarises the details of a particular stage such that the order of reading is top-to-bottom, left-to-right.*

Figure 3.4 aims to communicate the relationships between the output data produced by each stage and the input data of each subsequent stage. For example you'll notice that the output of segmentation feeds directly into the vectorisation stage and the vector shapes derived from vectorisation feed the projection stage.

Figure 3.5 clarifies how the input, algorithmic control arguments and outputs fit together to form a coherent stand alone reconstructive unit. The structure of the MAMMAL algorithm is best described as a sequential pipeline which results in a composite scene-graph formed of a set of different types of polygon-mesh.

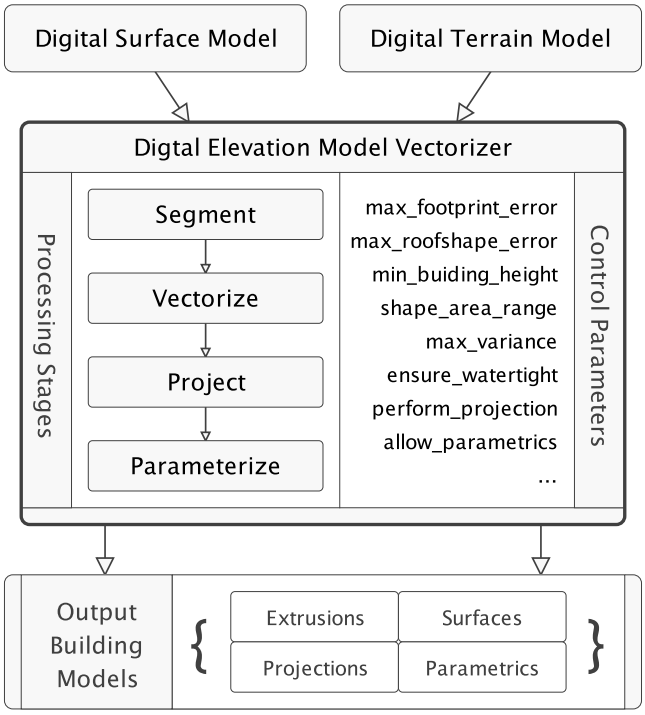

Figure 3.5: *schematic high-level overview of the proposed technique - the top row indicates the input data (dsm and dtm), the central block outlines the body of the method, and the last row the nature of the output scene-graph.*

Note: (regarding the use of regularly spaced gridded DEMs). The operator accepts structured digital elevation models as input. This decision was made for a number of reasons. Firstly for their availability. DEMs are readily accessible by many researchers and practitioners in the architectural and geometric communities - whilst access to city-scale unstructured scans is still somewhat exclusive. Secondly structured DEMs offer an incredibly efficient point storage mechanism, since for each point a single decimal value is required (the elevation) - whilst for unstructured scans, for each point, a minimum of 3 decimal values are required (the x, y, and z coordinates) with the potential for more (if surface-normal, point-intensity and RGB measures are also incorporated). Thirdly ESRI's digital elevation standard is well integrated with the existing GIS systems. Fourthly DEMs are increasingly becoming cheaper to acquire relative to the cost of unstructured airborne scans - with many providers (such as the Environments Agency) offering nationwide coverage freely for both academic and industrial use. Fifth, due to their regular gridded nature (a 2D matrix), structured DEMs are highly amenable to many of the pre-existing image-processing algorithms.

Pseudo-Code

Figure 3.6 provides a clearer outline of the procedural steps employed in order to turn the input airborne range scans into compact 3D building models. The following is a slightly more involved representation than the equivalent overview code provided in the previous chapter. This is simply because the algorithm is slightly

more involved than its predecessor. However the use of human-readable variable and function names aims to make the logic as readily digestible as is possible (given the complex nature of the problem). The main for loop is applied to each segmented building object in turn, in order to perform the vectorisation, projection and optimisation stages. The output (return value) scene-graph is composed incrementally as each segmented building is polygonised. The important aspect of this pseudo-code is the flow of information between the individual processes. It states which features of the input point-cloud are exploited to address each sub-task.

```
1  difference_of_elevations = digital_surface_model - digital_terrain_model
2  sliced = threshold(difference_of_elevations, min_building_height)
3  cc = connected_components(sliced)
4  cc = filter(cc, footprint_area_range, cellsize, max_variance)
5  models = {}
6  foreach (cluster in cc)
7      dense_footprint = linear_boundary_edge_traverse(cluster)
8      sparse_footprint = simplify_polygon(raw_footprint, max_footprint_error)
9      segments = maximal_area_segment(cluster, roofshape_area_range)
10     dense_roofshapes = linear_internal_edge_traverse(segments)
11     sparse_roofshapes = {}
12     if (ensure_watertight)
13         sparse_roofshapes = graph_refine(dense_roofshapes, max_roofshape_error)
14     else
15         foreach (polygon in dense_roofshapes)
16             rs = simplify_polygon(polygon, max_roofshape_error)
17             add(rs,sparse_roofshapes)
18     projection = project_2D3D(sparse_roofshapes, cluster, max_roofshape_error)
19     errors = depth_buffer_xz(projection, cluster)
20     if (enable_parametrics)
21         parametric = optimise(cluster,sparse_footprint,sparse_roofshapes,
22             max_footprint_error, max_roofshape_error, min_intersect_over_union)
23         paramerrors = depth_buffer_xz(parametric, cluster)
24         add(accept(mean_error(paramerror))) ? parametric : projection, models)
25     else add(projection,models)
26 return models
```

Figure 3.6: *high-level pseudo-code of the airborne mass reconstructor*

Once again type statements are omitted so as to ensure the code is generic and flexible. Interestingly, although this appears to be quite a high-level representation, it is actually a direct copy of the function MAMMAL, which was implemented in C++, C# and Java. Having read this you should have a clear idea of algorithm. The only thing that may be ambiguous is the precise nature of the sub-routines that are invoked (such as $graph\_refine()$, $optimise()$ and $maximal\_area\_segment()$). The explanations presented in the methodology section resolve these ambiguities. In truth an experienced engineer could easily reproduce a working variant of the algorithm, just from this description, however there are a number of critical factors, which (if overlooked) have the potential to degrade its performance. One of the key aims of this chapter is to demonstrate the manner in which a reasonably standard processing pipeline (segment, ..., polygonise) can be refactored using simple data-driven logics in order to both maximise the computational efficiency and to actively control the topology of output building models.

Finally before progressing to the body of the approach - the following figures depict

both the input data and the types of derived data that are exploited by the airborne mass-reconstruction algorithm defined in this chapter.

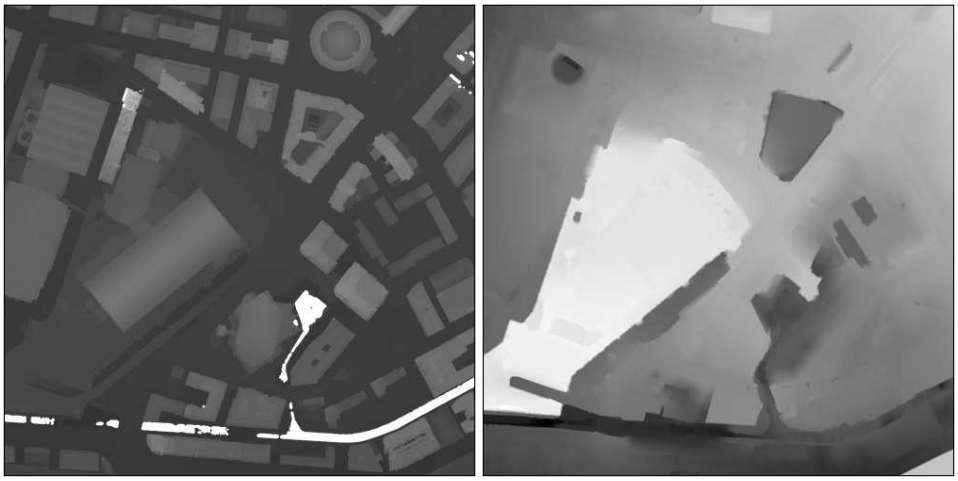

Figure 3.7: *a digital surface model (left) and digital terrain model (right) of a portion of the City of Manchester dataset - at 25cm point spacing - this illustrates one the datasets used during exposition of the MAMMAL algorithm, its analysis and its evaluation - it is rendered with the elevation at each point mapped to pixel intensity such that lighter areas correspond to higher altitudes and missing data is indicated with fully transparent pixels.*

Figure 3.7 illustrates the key pieces of input used by the aerial (top-down, 2.5D) building vectoriser. Figure 3.8 on the other hand illustrates a photograph of the region for qualitative comparison and a surface-normal map, computed by applying the Sobel operator to the digital surface model.

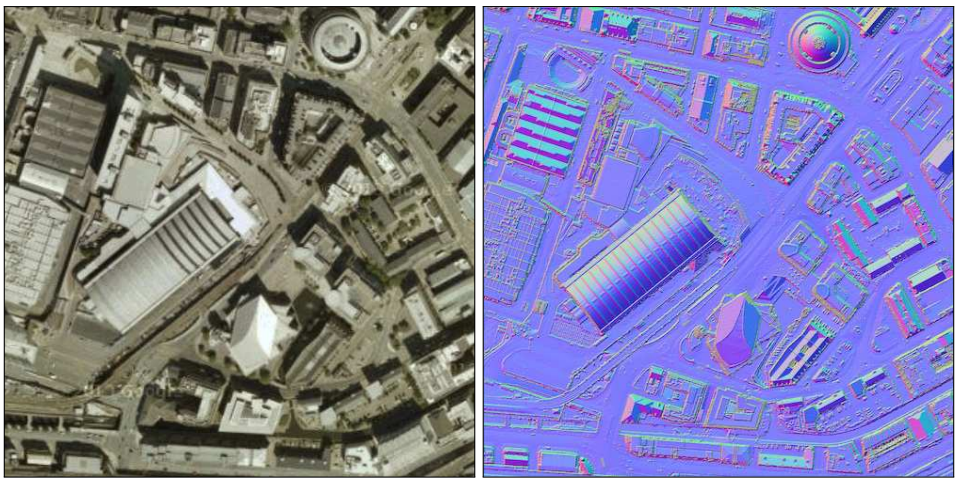

Figure 3.8: *an aerial photograph (left) and Sobel derivatives (right) for the region of the City of Manchester dataset indicated in figure 3.7 - the Sobel derivatives provide estimates of point-normals for the algorithm based on a pixel neighbour image-processing operation, whilst the aerial image (courtesy of Google maps) which is not used in geometry recovery, provides a visual representation that a human can use to perform qualitative comparisons.*

3.3.2 Segmentation

The first step in the MAMMAL algorithm is to break down the input point-cloud into salient clusters of points. To achieve this a two-stage segmentation method is applied to the input point-cloud. The first stage isolates individual buildings. The second stage identifies individual roof-shapes within each building. In order to segment each building, the difference of elevation models (exploited in the previous chapter) is first applied. In order to segment each building's constituent roof-shapes, a novel segmentation algorithm (named MARS) is exploited. The core idea of the MARS algorithm (Maximal-Area Roof-shape Segmentation), is to ensure that as-large-as-is-possible (given an error tolerance) segments are generated, by actively suppressing non-conformal clusters of points. The reasoning (as previously stated) is that larger output per-face surface-areas are both indicative of manually constructed CAD models and more efficient to polygonise. This initial segmentation stage is critical to enabling this in the sense that it is responsible for ensuring that the subsequent reconstructive stages are fed, clean semantically meaningful structural divisions of each building.

To support the explanation of the segmentation stage, a number of critical aspects are first considered, before the detailing of the two sub-tasks.

Foremost the input is a surface and terrain airborne range scan. The output is a set of integer identifier labels that map each point in the input surface scan to a distinct building, and a set of integer identifier labels that map each point in each building's point-cluster to a distinct roof-shape. In essence $S(dsm, dtm) \rightarrow \{B, R\}$: where S denotes the segmentation function, dsm and dtm (respectively) denote the surface and terrain DEMs, B maps to the output building labels (a 2D matrix of integers) and R to the output roof-shape labels (again a 2D matrix of integers). Where $|dsm| \equiv |dtm| \equiv |B| \equiv |R|$. Note: the convention used is to assign ground (terrain) or clutter points to group zero (0), such that values in the output segmentations greater than zero each map to a point in a salient region.

The next subsection provides a brief recap of the difference of elevation models building segmentation method defined in the previous chapter.

Difference of Elevation Models Building Segmentation

To segment building footprints the *difference-of-elevation-models* is calculated by subtracting the input terrain-model from the input surface-model. This yields a matrix of normalised scalar elevation values. The matrix is passed to a threshold function which omits elements with value less than the user-supplied minimum building height (in meters). This in turn yields a set of disjoint clusters of points, which are extracted using a connected-component routine. At a high-level this has the effect of removing all points close to the ground terrain, leaving behind points corresponding to architecture. Clusters are then filtered in order to remove any remaining vegetation, vehicles, street-furniture and clutter, using a linear classifier.

Recall (from the previous chapter) that the classification features were inspired by the tree-detection constructs of Chen et al. [15] and the covariance-analysis derived discriminant features of Zhou et al. [162]. The weight associated with each feature was initially learnt using a support-vector-machine and then manually refined by hand to enhance performance at varying resolutions. As pointed out by Zhou, this type of problem can be addressed with many other linear-discriminant

algorithms.

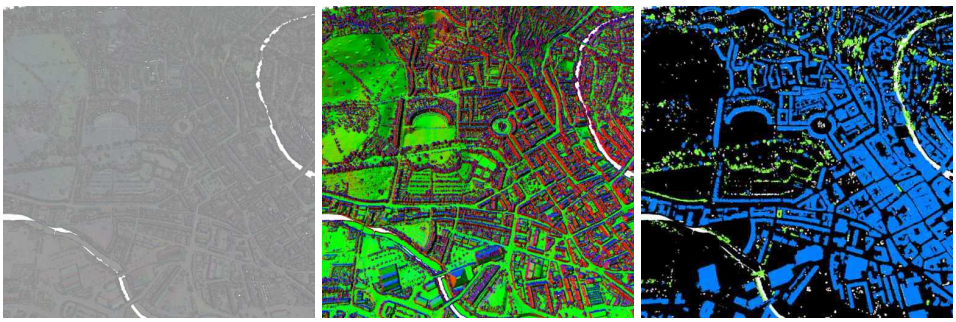

Figure 3.9: *a linear type classifier automatically filters manmade architectural elements from vegetation and clutter : (left) input points, (middle) intermediary derivatives and (right) class labels (blue=manmade)*

Figure 3.9 revises the result of the type-classifiers application to the city of Bath dataset. The right-hand image illustrates the segmented and classified objects, with blue regions denoting the building point-clusters and green the vegetation - the black points correspond to terrain and clutter.

Another point to recall is a subtlety of the classifier's behaviour. A key difference between the algorithm and its predecessors it that, although it computes point-level features, it actually classifies types at an object-level. In revision, the formula for the linear point-cluster type-classifier is:

$$f(A, man, \vec{w}) = \frac{(w_1 A_D + w_2 A_P) \times (1 - A_S)}{w_3 A_A + w_4 A_V} < 1 \tag{3.1}$$

where: A_D, A_P, A_S are scalars measuring the object's disparity, planarity and stability respectively. A_A and A_V refer to the objects footprint area and volume respectively. The influence of each component is controlled by the weights w_{1-4} of the vector $\vec{w}$. Intuitively (recall that) this means the greater the mass of an object the greater the likelihood it is manmade, whilst ensuring that as disparity increases and non-planarity dominates, the likelihood of an organic element rises. Inverting the stability measure ensures that complex or curved manmade objects, that exhibit significant continuity, are not mistakenly treated as organic.

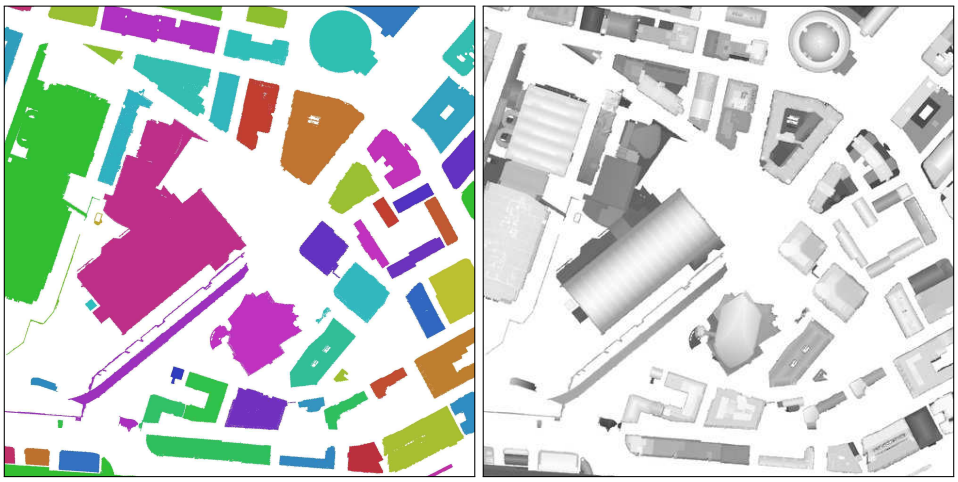

Figure 3.10: *DoEM clusters (left) with normalised building-clusters (right)*

Since the difference of elevation models building segmentation has already been detailed in the previous chapter, this recap is brief and does not repeat each of the implementation considerations. Readers that require a deeper outline should refer back to the previous chapter. Figure 3.10 illustrates the outcome of the difference of elevation models applied the subset of the city of Manchester dataset depicted in this section's overview. The image on the left displays the output building label matrix with each integer value mapped to a pseudo random colour. The right image displays the elevation of the original range points with the elevation-to-intensity colour mapping handled on a per-building-cluster basis (as a product of the segmentation). In particular you can already see the benefit of isolating individual objects before attempting to segment roof-shapes. In the image on the right the structure of each roof surface is represented more clearly than in the original digital surface model rendering (figure 3.7 - which has elevation values normalised globally for the whole dataset). This is another product of identifying salient regions in the input with connected-component extraction.

Based on the result of object segmentation and classification (depicted in figure 3.9), the MAMMAL algorithm then sub-segments each cluster of building points in order to localise on individual roof-shapes. This is handled by the newly-proposed MARS algorithm, which is discussed in detail next.

Maximal-Area Roof-Shape Segmentation

The aim of the MARS algorithm is to break down each disjoint cluster of building points into sets of roof-shape point clusters. For each cluster of building points P the goal is to label each of its points $p_i \in P$ as belonging to a single group: g. Such that the result of roof-shape segmentation is a mapping function, $M(P, p_i) \rightarrow g$, which describes the relationship between each point in a segmented building and the roof-shape it belongs to. This method computes the second segmentation mask R referred to earlier.

This subsection explains the three main components of the algorithm - which are graph-labelling, the application of connectivity constraints and non-conformal suppression of clusters that invalidate the maximal-area property.

Note that although the disjoint nature of building point-clusters makes it easy to extract connected-components during the *difference-of-elevation-models* stage, in order to effectively segment roof-shapes, a more involved technique is required, since neighbouring points (that are adjacent) may not belong to same roof-shape. Hence a graph-labelling strategy is employed. The benefits of the graph labelling over similar techniques (such as region-growing and quantization) are its efficiency (in terms of compute time and memory use) and the guarantee of a commutative return (since a bi-directional graph and connectivity predicates are used, the order of evaluation will not impact the formation of roof-shapes). The method is conceptually incredibly simple. The first step is to iterate over all the points in the input and for each individual point's neighbours determine whether a connective graph edge exists between the two. This results in the aforementioned bi-directional graph. Then connected graph components are extracted by taking the minimal spanning tree of each disjoint graph region. For structured data (such as the rasterised DEMs) there will be many minimal spanning trees for each region (since the distance between neighbouring points is constant) whilst for unstructured data there will be one or more. However in both cases the nature of the MST will not matter,

since it is only used to group contiguous sets of points. This is because the edges of the MST are not required for any further processing.

The positive aspect of this is that it will not matter which initial point is selected in the first round of grouping points since for any two points a simple predicate tests whether they belong to the same region. Simply put using the bi-directional graph of local point connections, two points belong to the same roof-shape if a path or edges exists between them. The simplicity of the strategy is not the only benefit. When an image-based DEM representation is used, each of these stages can be computed in linear time relative to the size of the input. This is because for each point a constant number of evaluations is necessary (which is controlled by the neighbourhood size - 4-way or 8-way). In the case of unstructured data however, although the logic is the same, the local connectivity tests will vary (in their execution time) proportionally to the number of points in each neighbourhood. This is controlled by the neighbour radius or count used for the lower level point-location queries. As many have pointed out [87], [52], [60] there is generally an exponential growth associated with larger support-radii in nearest-neighbour construction for 3D point-sets. This should explain the predominant use of discretized DEMs, since instead of having to traverse an oct-tree or volume (to speed up spatial queries), the algorithm can simply index into a regularly spaced grid representation. This a product of the parallax nature of airborne scans and generally is not applicable to arbitrary point-clouds.

Having outlined the graph labelling, the next detail is the connectivity predicates that are applied to individual pairs of points to test if they are contiguous. Three attributes of the input points are exploited to achieve this. These are the variance in: the elevation of points, the derivative of points (surface-normal) and the continuity of each neighbourhood (as a measure of curvature change). The following expressions formalise these predicates:

- Elevation: $f(p, q) = ||p_y - p_y|| < \alpha$
- Surface-Normal: $f(\vec{n_i}, \vec{n_j}) = (||\vec{n_i} - \vec{n_j}|| \times 0.5) < \beta$
- Curvature: $f(c_i, c_j) = ||c_i - c_j|| < \omega$

The elevation term considers the difference in height between points p and q using the height-change threshold α. The surface-normal term evaluates the magnitude of the difference vector between two unit normals n_i and n_j derived from a pair of point neighbourhoods with values less than β denoting region continuity. Note there is a non-linear relationship between the magnitude of the difference of the unit normals and their angular variance. One could alternatively use the distance between them on the Gaussian sphere. This simpler formulation presented relies on the fact that the greatest variance vector between two unit normals centred at the origin has magnitude two. The primary reason for this expression is compute speed relative to the spherical angular distance. The curvature term determines the difference between two points as a product of measuring the curvature and thresholding with ω as the largest acceptable deviance in curvature between two adjacent points that belong to the same region.

The application of these measures of *connectedness* in conjunction with the graph labelling routine results in a data-driven subdivision of each building's point-cluster, based solely on the local properties of points.

However, due to the nature of the input data (riddled with low and high frequency sensing noise) this stage generally results in an over-segmentation, since points that may actually belong to the same roof-patch may be treated as distinct as a result of the sampling artefacts. However this is not a problem and is actually key to the algorithm's efficacy. By over-segmenting in this low-level manner the operator ensures that each building point-cluster is broken down into stable but potentially verbose sets of regions. This enables the maximisation stage discussed subsequently. Note: the scope of the over-segmentation is controlled by the thresholds used during evaluation of the connectivity-predicates - which are specified by an end user.

The last stage in the MARS algorithm is to suppress roof-shape clusters that do not conform to the user supplied minimum roof-shape surface area. This key process ensures that less verbose segmentation masks are returned. It is vital to the notion of *maximising the mean surface-area* of roof-shapes. One can think of this as analogous to non-maximal-suppression in signals processing. Although here it is implemented in a geometric sense. In essence the aim is to refactor small, noisy roof-shapes that typically correspond to chimneys, aerials and air-conditioning units so as to yield cleaner, more compact segmentations. Further in an ideal world the refactoring routine should also minimise its impact on an output segmentation's structural accuracy - such that it acts simply as a scale-space feature suppressing filter.

Two iterative non-conformal suppression routines are exploited by the MARS algorithm. The first operates by incrementally merging small non-conformal clusters with their dominant conformal neighbouring roof-shapes. The second operates by merging small non-conformal clusters with other non-conformal neighbouring clusters in an attempt to ensure conformal clusters.

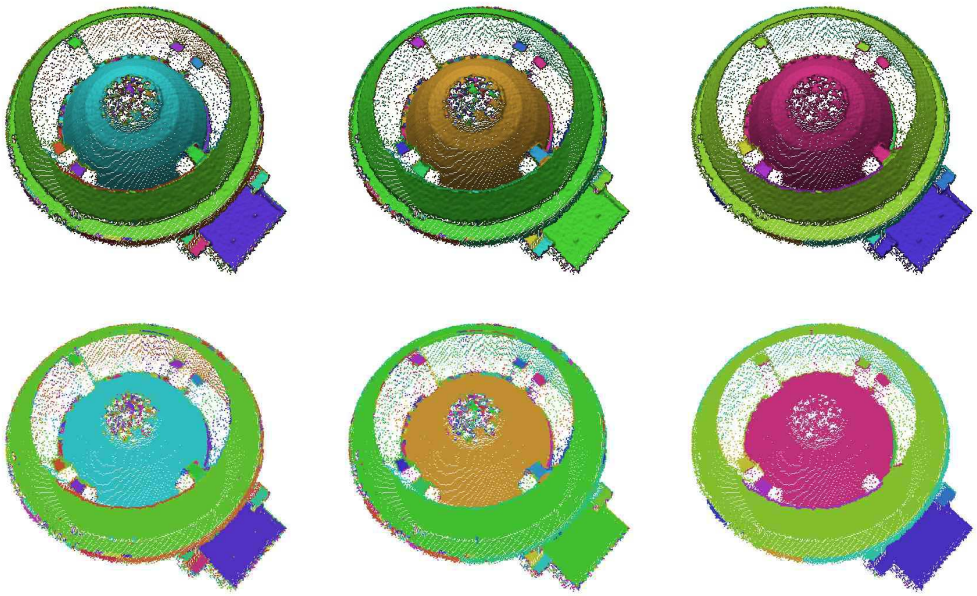

Figure 3.11: *MARS' suppression of non-conformal point-clusters significantly reduces the cardinality of the segmented set, whilst retaining the structure of each division - (top-row) vertex-lit, (bottom-row) unlit (for clarity)*

Figure 3.11 seeks to clarify the outcome of the non-conformal suppression stage of the MARS algorithm using the example of the Manchester public library. It depicts (on the left and center) the raw roof-shape segments (generated by applying the local-connectivity constraints and graph-labelling) and (on the right) the refactored roof-shape segments (generated by applying the iterative suppression methods described above). The top row displays the same result as the second with the difference in appearance resulting from toggling lighting. The factor that should be noted (in particular) is the removal of small noisy segments, such as those present in the middle of the central dome. If you zoom in you'll notice numerous roof-shapes have been refactored into a single representative roof-shape. The unlit row aims to make this more apparent. Although the change is subtle (when viewed at a distance), it has important implications for the polygonisation stages.

The core idea is that whilst the visual appearance of the refactored segmentations are equivalent from a human perspective, from a computational perspective, the cardinality of the set of roof-shape segments has been dramatically reduced, from thousands down to tens. The reason this is important is because the greater the number of roof-shapes, the increase in the latency of the vectoriser applied subsequently. Essentially an excess of small un-important clusters degrades the performance of the transition from points to vectors, both in terms of execution time and topological quality. The refactoring applied by the MARS algorithm not only enhances efficiency, but supports the key reconstructive principle exploited in this chapter, that of maximising the mean surface-area of roof-shape faces whilst conforming to a user-supplied error tolerance. Combining DoEM segmentation with MARS results in fully automatic building and roof-shape segmentation (figure 3.12).

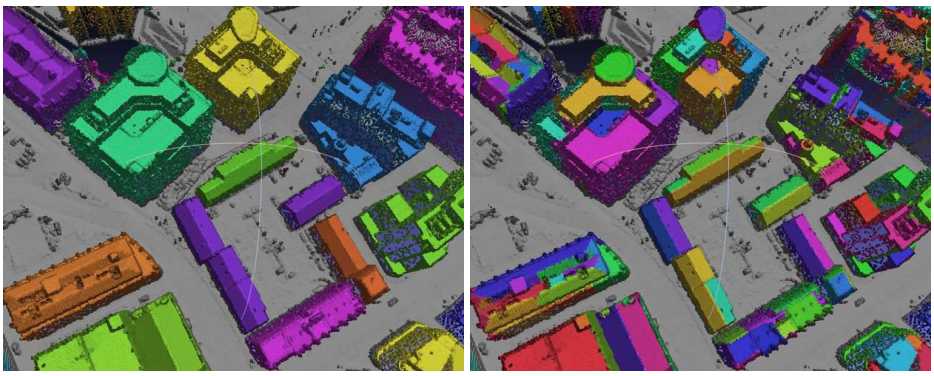

Figure 3.12: *roofshape segmentation identifies sub-components*

Figure 3.12 illustrates the outcome of the MAMMAL algorithm's segmentation on a subset of the Manchester dataset (depicted in this section's overview) in order to help clarify the behaviour described. The left-hand image depicts segmented building point-set clusters rendered with pseudo-random group colours. The right-hand image depicts each building's constituent roof-shapes again rendered with pseudo-random group colours. In both images the ground terrain and filtered clutter are indicated by gray points. The minimum footprint area for the left-hand result is set at 40m^2 whilst the minimum roof-shape area for the right result is set at 10m^2.

One other important aspect of this approach is the scope for an end user to vary the level of detail present in each building's roof-shape mask.

Continuous Scale-Space Level-of-Detail

A vital feature of the algorithm (relative to the pre-existing methods) is the ability to continuously vary the level of detail of the output projected models, by varying the input (user-supplied) roof-shape area range. Figure 3.13 illustrates this. The core idea is to do away with discrete level-of-detail strategies (as in [156]) in favour of a means to smoothly alter the LOD.

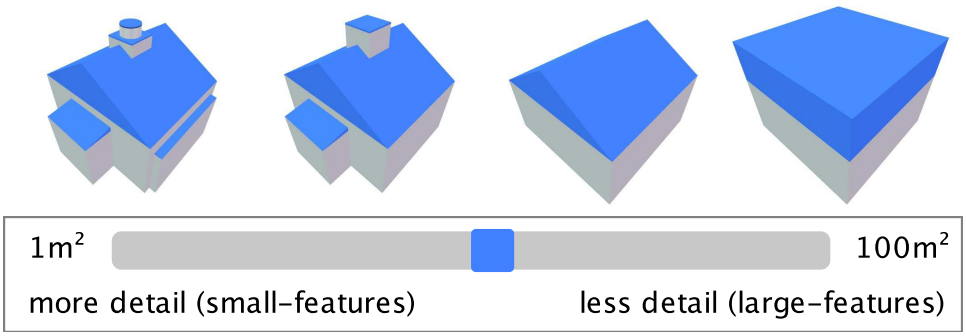

Figure 3.13: *continuous-level-of-detail segmentation enables the MARS segmentation algorithm to characterise each building at different scale-spaces*

The benefit of the approach is that it enables the *grain-of-features* present in reconstructed buildings to be controlled intuitively using a real-valued scale slider (in meter^2). This means an end-user can exploit the algorithm to recover both

accurate models (that capture precise roof-structures) and compact models (that mask insignificant objects on roofs), as well as a wide variety of options in between both extremes. The powerful thing about this approach is that it enables an end-user to control a key property that would otherwise have been pre-determined. This feature is driven by the non-conformal suppression stages of the MARS algorithm. Though it is important to note that although the method strives to resolve roof-shape segments so as to conform to the supplied minimum-roof-shape-area, the refactoring routine does not guarantee this. However practically (for all the experiments discussed in §3.4) the presence of non-conformal (small) roof-shape clusters did not impact the mean surface area of the resultant roof-shape faces detrimentally. In all cases the mean area still exceeded the user supplied minimum. Essentially the algorithm refactors as much as is feasible, based on the geometry of each building's roof and the stability, disparity and variance of the resultant merged roof-shapes. As further explanation of this the underlying reason is that in some instances, suppressing a small cluster of points can have a significant effect on the geometric accuracy of the output model, since for architectural elements such as spires (tapering conical or pyramidical structures), their omission can significantly alter the result. Ultimately there are particular types of roof-feature that (though small in area) cannot be suppressed without butchering the output building.

In summary, segmentation in the MAMMAL algorithm is a two-stage process that first identifies buildings (using the difference-of-elevation models method) and then identifies individual roof-shapes in each building (via a novel maximal-area roof-shape segmentation algorithm). The output of segmentation is fed directly to the vectorisation stage which is why it is important to ensure clean, structured divisions are generated. The key advantage of the proposed strategy over the pre-existing algorithms (such as [142], [163] and [31]) is the ability to handle both planarity and curvature in a data-driven manner. Additionally the method is computationally efficient (in terms of execution time) and supports continuous scale-space level of detail (as a means for end-users to control the *grain* of features present).

So having isolated salient clusters of points, the next stage in the MAMMAL algorithm is to polygonise each set of segmented points in order to yield 2D vector shapes for each detected building. Although one may wonder - why not simply generate 3D models once segmentation is complete? The answer is a result of a preference for computing in as low a dimension as is possible. Since dimension reduction is generally beneficial in terms of improving efficiency by reducing problem complexity. Additionally the recovery of 2D shapes (prior to 3D models) enhances the applicability of the MAMMAL algorithm to both 2D map-updating and 3D model-reconstruction.

3.3.3 Vectorisation

Vectorisation is the process that turns clusters of segmented points into vector shape data. MAMMAL achieves this via scan-conversion (that yields accurate dense vector-shapes from clusters of points) and polygon-simplification (which sparsely approximates each dense vector-shape). The method exploits a regular cartesian grid to extract extremal (footprint) boundaries and interior (roof-shape) boundaries from segmented point-clusters. Predicated on the 2.5D assumption this is achieved efficiently in two dimensions. The underlying insight is that the

structure of the parallax geometry can be characterised in a lower dimension and projected into R3 with minimal loss of information. The regular grid representation enables constant time point-location and neighbourhood queries which ensures the extraction of dense vector shapes executes in roughly linear time relative to the number of input points. Three key features are discussed next. The means of verifying accuracy in 2D, the approach to efficiently refining sets of co-related shapes and the algorithm's functional approach to 2D shape approximation.

Measuring Geometric Error in 2D

One of the key components of any vectorisation strategy is the method of calculating geometric error. The aim is to quantify the *fit* of an approximate shape relative to a target shape. The MAMMAL algorithm uses the discrete point-to-edge distance between a dense vector and a sparse approximate, coupled with the continuous intersection-over-union shape-matching test. They act as the basis for validating each sparse approximate vector shape. The point-to-edge distance (Hausdorff distance) is useful because it prevents points deviating from the target shapes boundary, whilst the shape overlap test ensures that the interior region of a shape is preserved.

Formally the aim of 2D polygonisation is to simultaneously minimise the Hausdorff distance between a point-clusters dense boundary and an approximating polygon, whilst maximising the intersection over the union of the region bounded by the polygon and the area represented by the point-cluster in 2D space. This can be stated algebraically with the following expressions:

$$min_x \in \mathbf{R} \;\; (x = f(A,B)) \;\; s.t. \;\; (A \cap B)/(A \cup B) \geq \alpha \tag{3.2}$$

$$f(A,B) = max(||A_i - (B_j, B_{j+1})||) \;\; \forall i \in A : \forall j \in B$$

where A and B are (respectively) the approximate and target shapes, $(A \cap B)/(A \cup B)$ is the intersection over the union of A and B, α is the minimum ratio (a scalar in the range [0:1]) and $||A_i - (B_j, B_{j+1})||$ is the geodesic distance between vertex i of A and edge $j \rightarrow j+1$ of B. You'll notice there is the trivial solution when A=B which is an undesirable outcome. Hence rather than seeking to find the globally optimum solution (which would mean returning the input unrefined) - the algorithm instead seeks to minimise the number of vertices in the approximate vector-shape under the constraint that $(A \cap B)/(A \cup B) \geq \alpha$ and $hausdorff(A,B) \leq err_{max}$.

The Hausdorff distance yields positive scalar values whilst the intersection over union test returns unit-less values in the range [0:1]. By combining the two expressions with the logical AND operation the effectiveness of the error measure is enhanced relative to exploiting only one of the measures in isolation. In essence the MAMMAL algorithm considers both the boundary and interior of a pair of shapes when determining the error between them. The primary reason is to improve the error measure's robustness. It also enables the introduction of heuristic approximation strategies whose return may deviate from the target shape since there is a stable means to determine how well the approximate fits. However before discussing the simplification strategies employed, the next block covers the process of scan-converting the segmented point-clusters in order to generate dense boundary representations of each building's ground-aligned footprint and constituent roof-shapes.

Scan-Converting Cartesian Grids for Efficient Boundary-Extraction

Before simplifying each 2D boundary representation, the MAMMAL algorithm must first determine an accurate (although verbose) boundary, that can be used as a *target* shape during simplification. As stated this is achieved by linearly traversing boundary edges in a discrete cartesian grid of region labels. Although there are a number of competing methods for 2D boundary extraction, the method of scan-conversion proposed is advantageous for a number of reasons. Figure 3.14 summarises the behavioural properties of the boundary extraction methods considered during development.

	Co-Linear Points	Concave Shapes	Internal Holes	Minimises Area	Fully-Automated	Complexity (big O)
Convex Hull	✓				✓	O(n log h)
Swinging Arm		✓			✓	O(log n)
K Nearest Neighbour		✓		✓		O(n log h)
Alpha Shape	✓	✓	✓	✓		$O(n^2)$
Edge Traversal	✓	✓	✓	✓	✓	O(n)
Point Traversal		✓	✓	✓	✓	O(n)

Figure 3.14: *comparison of 2D hull construction algorithms*

In figure 3.14, the properties in the header of the table indicate: whether the algorithm can deal with co-linear points in the input, whether the algorithm can represent concave shapes, whether the algorithm recovers interior (non-simplectic) hole boundaries, whether the algorithm returns an extremal hull that uses the minimal area required to represent the input, whether the algorithm can be exploited in a fully automated pipeline, and the complexity of the algorithm in terms of the growth in execution time as a product of the number of input points (n) and the number of output vertices (h).

In particular, note that although the $\alpha - shape$ supports the extraction of interior holes, one must first determine an effective α value (which is the radius of the circumscribed circle used to test for edges between points). This α value controls the extent of the simplification, however although it is intuitive to understand, it is non-trivial to estimate for arbitrary point-sets. The linear boundary edge traversal (LBET) approach on the other hand, meets each of the requirements set out in the table whilst also being highly efficient. One key difference between LBET and the alternatives (that must be noted) is that it relies on a discrete cartesian grid - since it is effectively an image processing operator. Further unlike the alternatives (that treat input points as entities with infinitesimal area) the LBET method explicitly exploits an area based representation in order to counter issues such as co-linearities and the potential for degenerate vector output.

Figure 3.15 seeks to clarify this subtlety by stepping through the processes of pixel-traversal and edge-traversal in order to illustrates the limitation of the former. The thing to take note of is that the pixel based method (top-row) eventually reaches a point for which there is no succeeding immediate neighbour (top-right-cell), whilst for the edge based method this does not occur. This is the key difference - even given a single point as input the edge-based method will return a non-degenerate boundary whilst all methods based on zero-area points will return a degenerate or null boundary.

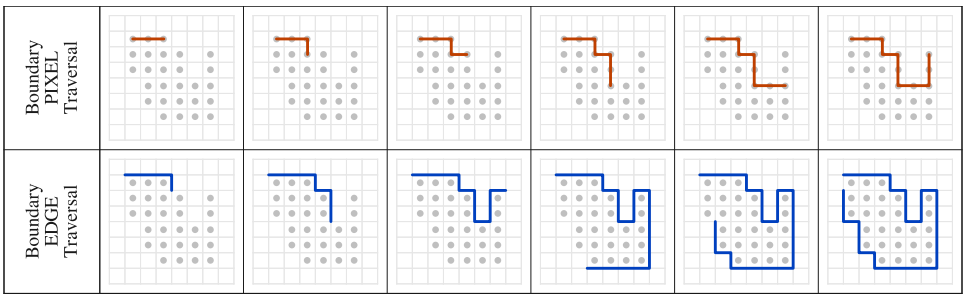

Figure 3.15: *Boundary **Pixel** Traversal against Boundary **Edge** Traversal of a cluster of points. Note the problems that can occur when the sample-points are treated as if they have infinitesimal area : (top-right-cell) traversal breaks due to the lack of a succeeding neighbour point*

One other vital feature of the dense LBET hulls is that they provide vector representations that can be used to perfectly reconstruct the 2D boundaries of each of the clusters of segmented points (without the loss of information) using a discrete grid. This property, coupled with the non-iterative nature of the algorithm make it ideal for fast and accurate vectorisation. However the issue is that they are heavy-weight descriptors and not immediately suited to visualisation. The next processes in vectorisation seek to transform the dense LBET hulls into edge-length maximising compact vector shapes.

To revise, the MAMMAL algorithm extracts dense boundary representations for clusters of segmented points prior to 2D shape simplification, using the linear boundary edge traversal method. The heavy-weight vector descriptors that result act as the target-shapes during the approximation process.

Graph-Based Topology Refinement in 2D

Graph-based topology refinement deterministically optimises the dense vectorised roof-shapes. The key difference between the algorithm's approach to refinement is that unlike shape-approximation, the result of graph refinement is a strict subset of the input dense points. This aspect is vital. In essence each graph refinement is a direct product of the input roof-shapes. Figure 3.16 illustrates this. You'll notice that the approximate footprint (in red) includes vertices that are not present in the dense footprint, whilst the graph refinement roof-shapes are constructed using only the points present.

The core aim of the graph-based topology refinement is to provide an error-bounded simplified version of the dense LBET hulls recovered during boundary extraction. The reason simplification is necessary is that it significantly reduces the number of vertices used to represent each hull, which not only minimises subsequent computation time but also enhances the visual appearance of the 2D vector shapes used to characterise each building on plan.

The input to the graph-refinement is a set of co-related dense-vector shapes (recovered by LBET) and the output is a set of co-related sparse-vector shapes. The underlying idea is to rely on the association of dense paths between the boundaries of neighbouring roof-shapes. The MAMMAL algorithm effectively *anchors* the simplification process at the ends of these shared-paths such that each roof-shape can be simplified by simplifying a sequence of paths, whilst maintaining the

manifold nature of the dense input. In essence the goal is to refactor each set of roof-shapes whilst ensuring that each transformed 'net' of roof-shapes is consistent and free of gaps and cracks. This approach relies upon the observation that for discrete integer masks one only needs to ensure that shared edges between neighbouring shapes are simplified in a commutative manner in order to ensure a watertight return. The commutative constraint is required because there is no guarantee that the ordering of the vertices in shared paths between neighbouring shapes will be the same. The implementation of this rather simple. First compute keypoints (as pixel intersections that make contact with three or more distinct roof-shape hulls - based on a regular cartesian grid). Then for each dense roof-shape, split it into a set of open paths, about the vertices that correspond to the anchoring keypoints. For each open path (in each roof-shape) apply a commutative (independent of order) simplification function to reduce the number of vertices. Finally (for each roof-shape) merge the set of simplified open-paths to yield the graph-refined hull.

The approach has 3 key benefits over existing methods of approximation:

- Direct and Unconstrained - in the sense that this function represents the data present without resort to fitting or sampling or constraint-based strategies. The output is purely a product of the input dense-hulls and the maximum (user-supplied) roof-shape error-tolerance.
- Embodiment of Topology - in the sense that shape neighbourhoods (in particular shared edges/paths that exist between neighbouring shapes) are preserved (remain consistent) during simplification. This means if a manifold shape-net is supplied as input then a manifold shape-net will be returned as output by the function. This is a useful benefit.
- Computational Efficiency - since this represents a non-parametric function - it is also non iterative to compute - in the sense it is invoked once for each building's vectorised dense roof-shape net. Furthermore the sub-process of identifying (and splitting about) anchoring keypoints actually has the unintentional benefit of speeding up each roof-shape's simplification, by simply breaking each down into smaller more efficiently processed dense-paths. Additionally it is possible to improve further by only simplifying the first instance of a pair of shared paths and simply propagating the first unto the second. However for this one must be careful to preserve the correct winding order of the duplicate.

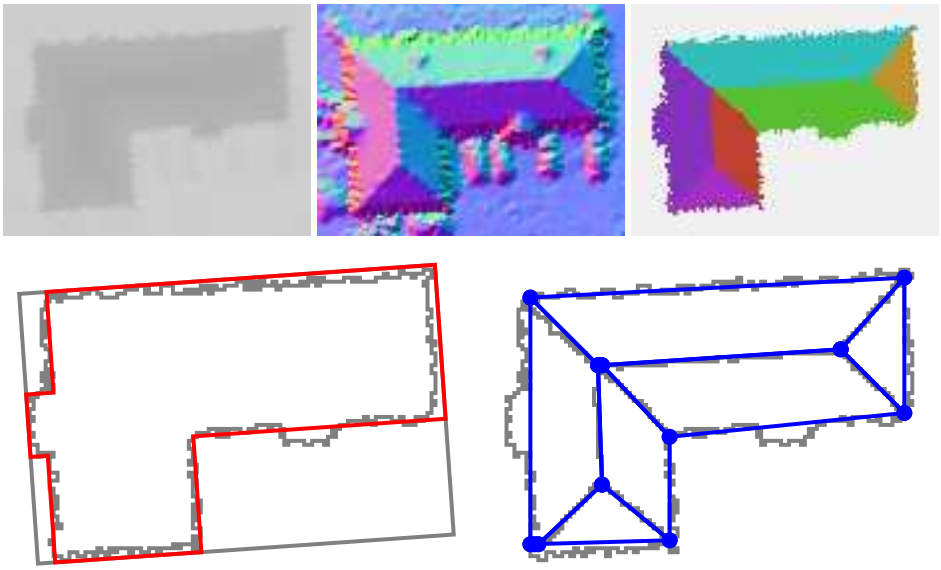

Figure 3.16: *the process of 2D graph refinement vs heuristic shape approximation at 25cm point-spacing illustrating (from left to right, top to base) input-points, normals, segmentation labels, footprint (dense in gray, approximate in red) and finally roofshapes (dense in gray, shape-graph in blue)*

Fundamentally, the graph-based topology refinement is a data-driven approach to simplifying sets of co-related shapes in two dimensions that guarantees to preserve the manifold nature of the input dense shapes. However because it is data-driven it does not suppress undesirable artefacts that may result such as spikes and significant perturbations in boundaries. This makes the previously discussed segmentation stage vital. Since this can only simplify that which it is given, if dense-hulls for un-refactored segmentations are supplied as input, they are preserved in the output. Although one could add a post processing step that suppresses such features, the result would be reducing the accuracy of the simplification and potentially invalidating the manifold property of the returned sparse-hulls. In some instances this may not be a problem, and so rather than force an end-user to use manifold shape-nets, the MAMMAL algorithm provides a switching flag that controls whether or not the output should be watertight. If watertight vectors are requested by an end user, then the graph-refinement occurs precisely as described. However if the user does not require watertight models, a further shape suppression stage is applied (similar to the non-conformal suppression used by the MARS algorithm), which further enhances the visual appearance of each sparse-hull but may invalidate the manifoldness of the return.

Figure 3.16 illustrates this process using a building from the Manchester dataset. Again the vital thing to note, is the difference between the approximate hull (base-left), and the graph-refined hulls (base-right). The graph-refined hulls are a strict subset of the input dense-hulls (in gray) whilst the approximate introduces new vertices that are not present in the input.

Note though, that graph-refinement does not enforce any topological or geometric priors. This allows effective data-driven characterisation of a set of shapes, however it also means that there is no enforcement of right-angles, edge-length

regularity or shape-symmetry. Fundamentally the graph-refinement aims only to efficiently characterise the data-present. In some instances however, it is desirable for an algorithm to introduce new vertices in order to enforce constraints on the geometry returned. Rather than fit lines (as [129] and [73]) the operator exploits 2D shape approximation functions.

One such approximation function (the hough-eat-away-hull) is illustrated in figure 3.16. The next subsection outlines these functions in greater detail.

Shape Approximation in 2D

The MAMMAL algorithm exploits 2D shape approximation functions to further enhance the visual and structural quality of each building's sparse 2D vector shapes. The core aim is to enable constraints and heuristic priors about shape arrangements to be enforced. In essence these functions allow highly specialised shape detectors and common shape detectors to be embedded within the algorithm's reconstructive logic. The primary reason is suitability for interactive simulation and visualisation - where compact (edge-length maximising) regularised shapes often appear to embody greater semantic meaning than jagged or perturbed precise sparse shapes.

The challenge here lies in striking a balance between how aggressively the algorithm approximates and the reduction in geometric accuracy that will result. Since any alteration to the sparse vector shapes will invariably alter the error properties, it is vital that such alterations are rigidly controlled. To achieve this the dual error measure introduced in §3.3.3 is exploited. It ensures that as the algorithm attempts to identify a high-quality approximate for each space vector shape, the associated error remains within globally set (user-supplied) tolerances. For reference (by default) MAMMAL uses a maximum point-to-edge distance of 1.5m and a minimum intersection-over-union ratio of 0.8 (80%). The remainder of this subsection, explains the behavioural attributes that are common to each of the 2D approximation functions and details two novel detectors developed during this project.

The key feature of each shape approximating function is a deterministic return. Unlike [142], [145], [73] and [1], the approach taken omits stochastic search, in favour of simple domain-specific 2D polygonisers. The rationale behind this is that is it preferable to extend and refactor pre-existing feature detectors to suit the purpose than it is to propose difficult to analytically characterise random sampling strategies. Although this is not the prevailing norm in architectural reconstruction (where derivatives of the RANSAC paradigm are bountiful), in this project the decision was made to favour deterministic methods over stochastic ones. This was largely a result of earlier experiments with multi-view stereo reconstruction algorithms (prior to dealing with actively sensed point-clouds). The key limitation of the MVS methods experimented with, lay in an inability to guarantee the same result given the same input. Although the differences were often subtle, it often proved necessary to repeat the reconstructive process multiple times and select the best result. This (as one can imagine) can be quite frustrating and (worse) wastes both time and computing power. Although one may accept such short-comings from techniques that derive geometry from images (for which there is no direct relationship between pixel intensity and the underlying object's surface), in the case of actively sensed point-clouds, this behaviour could be considered needlessly costly, since the difficult problem associated with photogrammetry (depth recovery) has already been

addressed. As a result - the notion of exploiting principled logics instead of random sampling is one that runs through the heart of this thesis. The 2D shape approximation functions developed stand as evidence of this.

There are two basic flavours of function exploited. Fixed form functions (alternatively referred to as common-shape-detectors) and open functions (also referred to as specialised-shape-detectors). Note: this distinction between two key types of approximation function is also relevant to the 3D parametric optimisation applied by the algorithm (discussed later). To help clarify, two simple shape detectors are introduced. The first is QUALM (for Quick Unconstrained Approximate L-Shape Method). The second is GRAILS (for Graph Refined Approximate Interior Linear Spine).

QUALM

This shape detector is an extension to the hough-transform and extracts rectilinear L, T and S shapes by first computing a minimal-area bounding box (MABB) and then *eats-away* at the corners of the MABB in order to characterise common architectural shapes. Figure 3.17 depicts the stages employed by the quick unconstrained approximate l-shape method. Note that the input can be either a structured or unstructured set of 2D points. The first stage computes the MABB, whilst the second stage refactors corners based on the distance from each MABB vertex to the input points. The aspect that makes this detector unique (amongst heuristic architectural footprint detectors) is its simplicity. It is conceptually incredibly intuitive and whilst it is heuristic in nature, it degrades gracefully at low resolutions. Beyond simplicity, there are additional benefits relative to pre-existing methods.

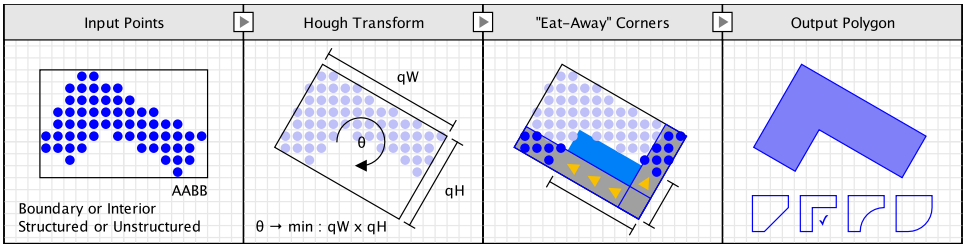

Figure 3.17: *Overview of the simple 2D shape approximation function - QUALM illustrating: (from left to right) the input points, the minimal area bounding box, then reducing error by 'eating-away' corners, and finally the output polygon (with alternative eat-away corner types illustrated below).*

Typically [134], fitting T and L shapes is achieved via analysis of a skeletal descriptor. The prevalent problem with these approaches is sensitivity to noise (perturbations) along shape boundaries [33]. One of the key benefits of the *hough-eataway-hull* is robustness to such noise. In particular you'll notice that in figures 3.18 and 3.19 the approximate hull is guaranteed to be composed of twelve or fewer vertices, irrespective of the undulations present at a shape's boundary. Whilst this makes it suitable for L,T and S shape detection, it also limits its utility as a general purpose shape approximation method. In this sense, the enhancement it provides for a subset of specialised cases comes at the cost of generality. However, given the frequency with which such common forms occur in urban environments, the utility of the heuristic is significant. Figures 3.18 and 3.19 depict the outcome of extracting eat-aways hulls for a handful of buildings sampled at 1m point-spacing

and 25cm point-spacing. Although the compact nature of the output hulls render them ideal for 2D map updating, before accepting an approximating eat-away hull, the MAMMAL algorithm first checks that it meets the user supplied error tolerance (using the Hausdorff and intersect-over-union measure outlined in §3.3.3). This ensures that the only 2D vector shapes that are optimised with this method are valid L, T or S shapes.

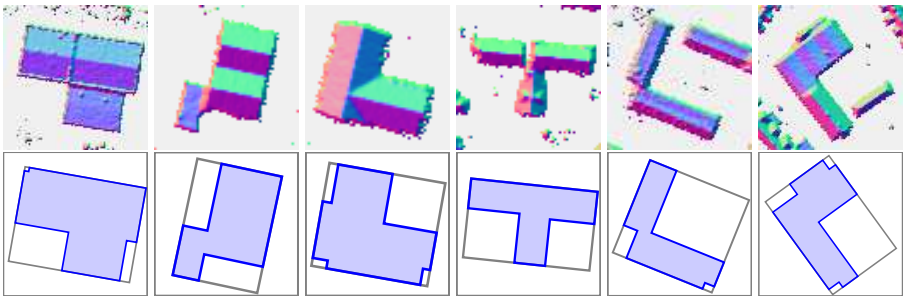

Figure 3.18: *building footprints automatically recovered from 1m point-spacing airborne range scans of the city of Bath, UK*

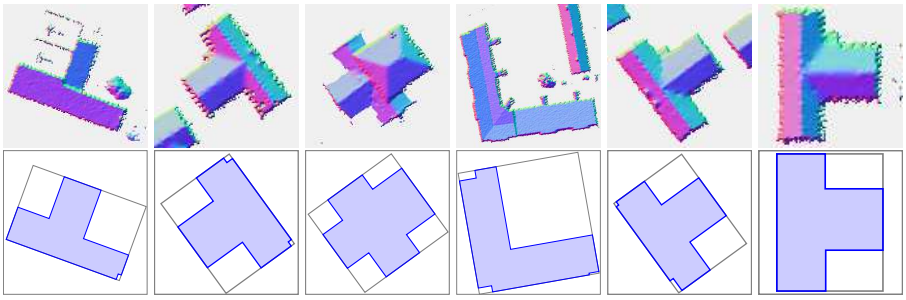

Figure 3.19: *Building footprints automatically recovered from 25cm point-spacing airborne range scans of the city of Manchester, UK*

One important note is that fixed-form functions such as QUALM can only be effectively applied to very particular classes of shape. This is essentially the key limitation of fixed-form functions such as QUALM. They deal very well with the type of data they were designed for, but fail to generalise to arbitrary input. The MAMMAL algorithm also exploits a handful of pre-existing 2D shape detectors to represent common shapes. These include a much simpler oval (circular-arc) detector, a quadrilateral detector and a regular-polygon detector. The addition of these common shape detectors enables the approximation of large portions of urban datasets efficiently. However as noted the fixed-form shape-detection functions generally fail to characterise architectural elements such as terraces, complexes, irregular and non-euclidean masses. The reason is that the fixed form functions are largely predetermined, and though they derive some of their properties from an input point-cluster, they only aim to represent one class of shape. For example a quadrilateral detector will not suddenly alter its return value in order to better deal with a concave footprint. It will always return a quadrilateral. In essence while fixed form functions are good at dealing with frequently occurring types of shape (that can be described formally ahead-of-time), in and of themselves they do not provide enough flexibility to yield high-quality (accurate and sparse) vector representations of arbitrary set of points.

For this reason a number of data-driven 2D shape detectors are used to address the limitations of fixed-form shape approximations. In particular, a medial-axis based railed terrace-detector (GRAILS) is discussed next.

GRAILS

GRAILS (graph-refined approximate interior linear spine) is a low level feature detector for 2D geometry whose purpose is to support the extraction of architectural features such as terraces and railed buildings. In essence this function computes as piece-wise linear spine given a dense shape boundary. The algorithm discussed is loosely related to (but distinct from) conventional shape *skeletonisation* methods such as the straight-line-skeleton [135], [2].

Although many algorithms tend to the problems of skeletonisation and (its counterpart) *boundary-extraction*, by comparison fewer formalisms exist for spine detection. This may in part be a result of the difficulty in defining the goal of such methods. The dominant strategy however is to erode or dilate an internal skeletal representation. The critical issue with such strategies is the requirement to iteratively refactor a dense representation. This can be computationally expensive, and worse can fail to effectively characterise a spine in the presence of noise at a shape's boundary. [134]

Recall from the preceding discussion of QUALM, that one of the tricky problems with refactoring the medial-axis of a shape (for skeletal template fitting) is sensitivity to sensing noise. The aim of GRAILS is to define a spine-extraction operator that is robust to sensing noise, computationally efficient and that generalises well to different types of geometric data. In this sense it should not be constrained to monotonic polygons for example.

The key idea is to treat the problem of spine-detection as the task of identifying the *longest-length non-cyclical path of a skeletal bi-directional graph*. In essence the algorithm takes (as input) a polygonal simplex (in 2D) or volumetric representation (in 3D) and generates an open sequence of interior vertices which represent the longest non-self-intersecting path amongst the medial-axis keypoints (which are the loci of the maximally inscribed discs) of the input geometry. The main application of the algorithm is in data-driven procedural reconstruction of architectural geometry from laser-scanned point-clouds. However due to its generality GRAILS has potential to be exploited within other areas of active sensing and computer vision.

The steps taken by the GRAILS function are incredibly simple to understand. For each building first compute the medial-axis (a tree-like structure composed of straight-line-segments and parabolic arcs). Based on the medial axis construct a graph in which the loci of a building's maximally inscribed discs are treated as nodes and the interior shared-edges of the Voronoi diagram of the building's boundary are graph-edges. Then compute the minimum spanning tree of this graph and use Dijkstra's algorithm to identify the maximal length path between any two nodes in the graph. Use the start and end nodes of the maximal length graph-path as the start and end positions of the GRAILS spine. Finally (in order to turn the simplified spine into a closed shape) compute the mean distance to the dense boundary and use this as the rail projection distance (the 'radius' of the rail). Essentially: compute the medial axis of the input, then compute the maximal-length, non-cyclical path of the graph of medial axis locus points and the resulting

sequence of vertices is a graph refined approximate interior linear spine.

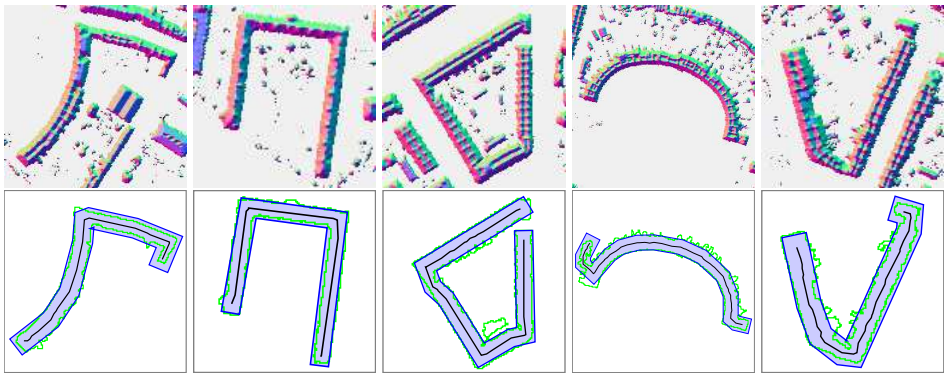

Figure 3.20: *computing sparse medial-axis derived 'rails' from 1m point-spacing terraced buildings in the city-of-Bath dataset*

To help clarify what is meant by a graph-refined interior linear spine, figure 3.20 illustrates the outcome of the 2D terrace detector exploited by the MAMMAL algorithm. Figure 3.20 depicts dense input shape boundaries in green, the medial axis paths in black, and the simplified rails (derived from the medial axis paths) in blue - for a set of irregular terraced buildings from the city of Bath dataset at 1m point-spacing. The thing to note about the outcome is that it freely adapts to the input data, unlike the fixed-form functions which attempt to mould a pre-defined template to the data. This is vital since it allows GRAILS to simultaneously approximate shapes made of straight line segments and curvature in a non-iterative manner.

Readers interested in implementing GRAILS (as a general purpose geometric feature detector) should refer to this chapters appendix for the full implementation. Special attention is given to it here because (to the author's knowledge) it is a novel algorithm whose properties make it highly suited to procedural reconstruction of architecture. Additionally from a geometric perspective it is quite an interesting approach since it is not bound to two dimensions, but can be extended to 3D spine detection. Note: GRAILS also plays an important role in the 3D optimisation discussed later.

In total three data-driven specialised shape detectors are exploited by the MAMMAL algorithm. Alongside GRAILS, a variant of the hough-eataway hull (designed for arbitrary simple polygons) and a greedy edge-length maximising simplification routine are vital to being able to tackle the problem of general purpose shape approximation in two dimensions. The thing to remember through-out is that the construction of the specialised (or data-driven) hulls is controlled by the input, whilst the common (or fixed-form) hulls are more akin to template-based approximation strategies.

To summarise, the vectorisation stage employed by the MAMMAL algorithm transforms clusters of segmented points into sparse 2D vector shapes. This is achieved by first scan-converting discrete cartesian grids, then simplification using the data-driven manifold preserving operator GROVE (graph refinement operator for vector extraction) and finally approximation using two types of deterministic shape detection function (template-driven such as QUALM, and data-driven as in GRAILS). MAMMAL exploits the dual error measure explained in section 3.3.3 to ensure that

during approximation any alterations to each building's boundary descriptors are error-bounded.

Once vector extraction, sparsification and approximation are complete the MAMMAL algorithm then creates 3D surface geometry from the 2D shapes so as to match the input laser-scan points for each segmented building. The next section discusses this stage - the projection from R2 to R3.

3.3.4 Projection

Projection is the means via which the 2D polygonal shapes (which are recovered during vectorisation) are positioned in 3D space so that they accurately match the original segmented laser scan points. Formally the aim of the projection stage is to minimise the RMS error between each projected vector shape and the original cluster of 3D points used in its construction (as demonstrated by figure 3.21). The projection routines exploited by the MAMMAL algorithm all take advantage of the parallax nature of the input airborne LiDAR which ensures that the geometric variance can be modelled as a product of simply altering the elevation of roof-shapes. A good analogy for this stage is *popping-up* a 2D cut-out. In essence MAMMAL pushes the 2D shapes up into 3D based on the position of the segmented points.

More formally the algorithm determines a mapping from R2 to R3 such that the RMS error between each resultant 3D surface and its constituent input points, is minimal. Once again it is vital to understand that this is only applicable because of the 2.5D nature of the input. In the case of generic unstructured point-clouds (such as ground-based laser scans), this approach will generally lead to the loss of information.

There are only really two options for this sort of projection - linear or non-linear. Linear projections are composed entirely of planar roof-shapes, whilst non-linear projections are the result of irregularities and curvature. The remainder of this section covers both of these options.

Projecting Roof-Shapes from 2D into 3D

Before detailing the linear and non-linear projections, a quick primer on the methods of calculating roof-shape elevations from sets of sampled points is provided. During experimentation, various low-level routines for calculating a vertex's offset position were considered (and are outlined below).

- Direct-Point-Elevation: which simply indexes into the original structured DEM array using the longitude and latitude of a vertex and returns the elevation at that position. The key limitation is a quantizing effect which is noticeable as discontinuities and the lack of a filtering strategy to omit elevations resulting in invalid elevations. The benefit of this point-level method is compute speed.
- Bilinearly-Interpolated Elevation: which extends the direct-point elevation method by bilinearly interpolating between neighbouring points in the DEM in order to limit the discontinuities (that result from the quantized indexing) whilst maintaining efficiency. However this still does not prevent invalid elevations, and can also (as the direct point elevation) lead to roof-shapes that should be planar, being modelled as non-planar polygons.

- Distinct Plane Estimate Elevation: which calculates the projected height of each roof-shape vertex by estimating a single plane for each roof-shape and offsetting each of its vertex based on the elevation of the plane at the longitude and latitude of the vertex. Although this is a more robust approach (relative to the point-based projections) the downside is that although discontinuities will no longer occur within a single roof-shape, across neighbouring roof-shapes, discontinuities can still occur in the elevation of shared edges.
- Quadratic-Error-Minimiser Elevation: which extends the plane-estimate elevation by considering all roof-shapes adjacent to a vertex in order to prevent variances in elevation at shared roof-shape boundaries by computing planes for each adjacent roof-shape and calculating a vertex's offset by minimising the quadratic error between each neighbour. Although this prevents discontinuities, the topological control comes at the expense of computational efficiency, since for each vertex, the error minimiser elevation must be estimated, that takes into account all of the shapes that are neighbours of the vertex. This is somewhat similar to Zhou's method of calculating vertex positions during 2.5D dual contouring however his underlying data-structure is a hermite grid, whilst here a graph of co-related polygon shapes is exploited.

Each of these low-level routines has its own advantages and disadvantages. In particular the Direct-Point Elevation provides a (computationally) cheap and efficient estimate and although the Bilinearly-Interpolated Elevation is slightly better both are still only point level projection methods. The Distinct Plane Estimate Elevation is better still, but it is not perfect because it does not consider neighbouring roof-shapes. Yet although the Quadratic-Error-Minimiser Elevation works quite well for certain classes of building roof (pitched, sloping, SLS-style), in many cases it performs worse than the Distinct Plane Estimate, since it assumes that all shared edges must be continuous, which yields erroneous results for *stepped roof-edges*.

Fundamentally it is worth noting that there is no single optimal parallax projection method for roof-shapes. The good thing about the point-level routines, is that for well sampled and cleanly segmented pointsets, there resultant accuracy will generally surpass the region level methods since they are derived directly from the points. However as the proportion contributed by noise increases the benefits of the region based projection routines starts to outweigh the directness of the point-level methods, because they effectively mitigate the presence of anomalous points.

Planar Projections

In the case of projecting linear roof-shape components - the problem is equivalent to finding the best approximating plane passing through a set of points - for which one can simply take the the least-squares plane or the smallest eigen-vector resulting from principle component analysis (PCA). The MAMMAL algorithm opts for the latter since it is less sensitive to outliers and sensing noise. However typically both approaches yield stable results since the segmentation stage isolates groups of points whose inter-group disparity is low and stability high. Figure 3.21 illustrates the outcome of a linear roof-shape projection applied to a building from the 25cm point spacing city of Manchester dataset to clarify. The left hand-side depicts the input points rendered with each point's normal mapped to a colour. The central

image shows the mean height-extrusions, whilst the right hand image depicts the building model attained by applying the PCA plane projection.

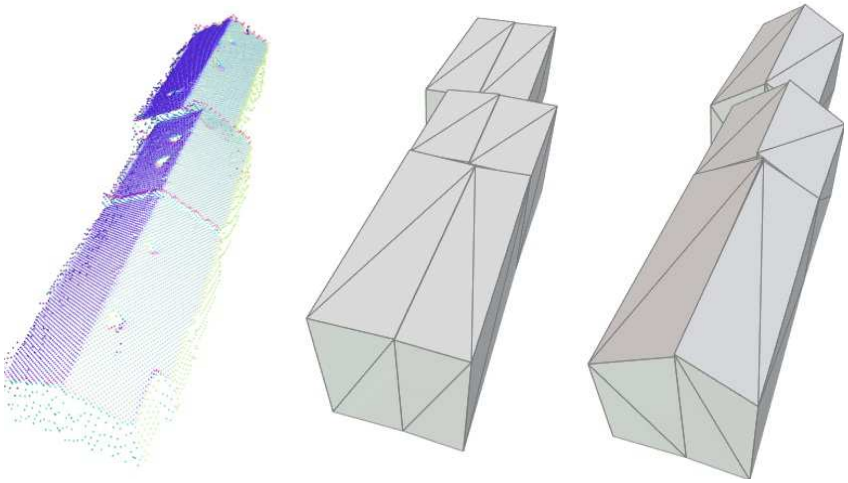

Figure 3.21: *an example of an unrefined linear projection from the Manchester dataset at 25cm point spacing - illustrating (from left to right): the input-points, the mean-height-extrusions and the projected-roof-shape-model*

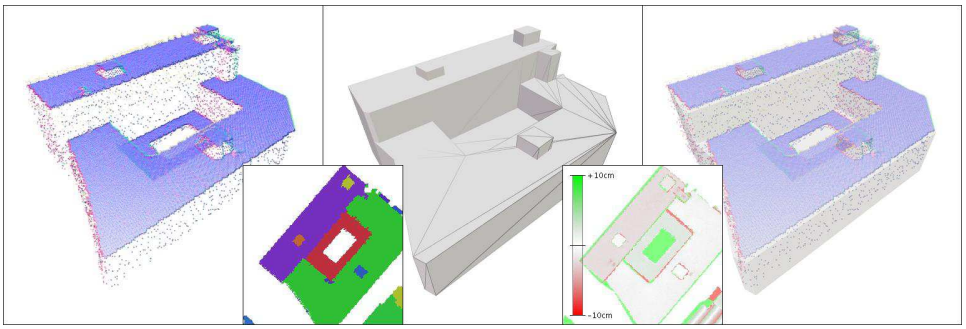

Figure 3.22: *a building characterised using flat roof-shape extrusions - illustrating (from left to right): normal rendered points, segmentation mask, extruded model, signed error matrix and model overlain with input-points*

As an interesting side note, during experimentation a special simple case of planar projections was identified. Flat (constant-height) roof-shapes occur quite often in urban environments, and although they are not applicable to all building roofs, they provide a fast initial projection method that ensures flat roof-shapes are not mistakenly modelled by slanted faces as a result of the presence of sensing noise. Essentially some building instances are best characterised as combinations of regular extrusions. Figure 3.22 clarifies this using another example building from the Manchester dataset.

Non-Planar Projections

Non-linear shape projections are required to explicitly represent curved and irregular building components. In such cases the algorithm computes structured polygonal shape divisions in order to control the distribution of vertices across each

roof-shape. The method then projects the sub-divided 2D shape set into 3D by minimising the quadratic error between each vertex and the set of planes derived from faces with edges adjacent to the vertex.

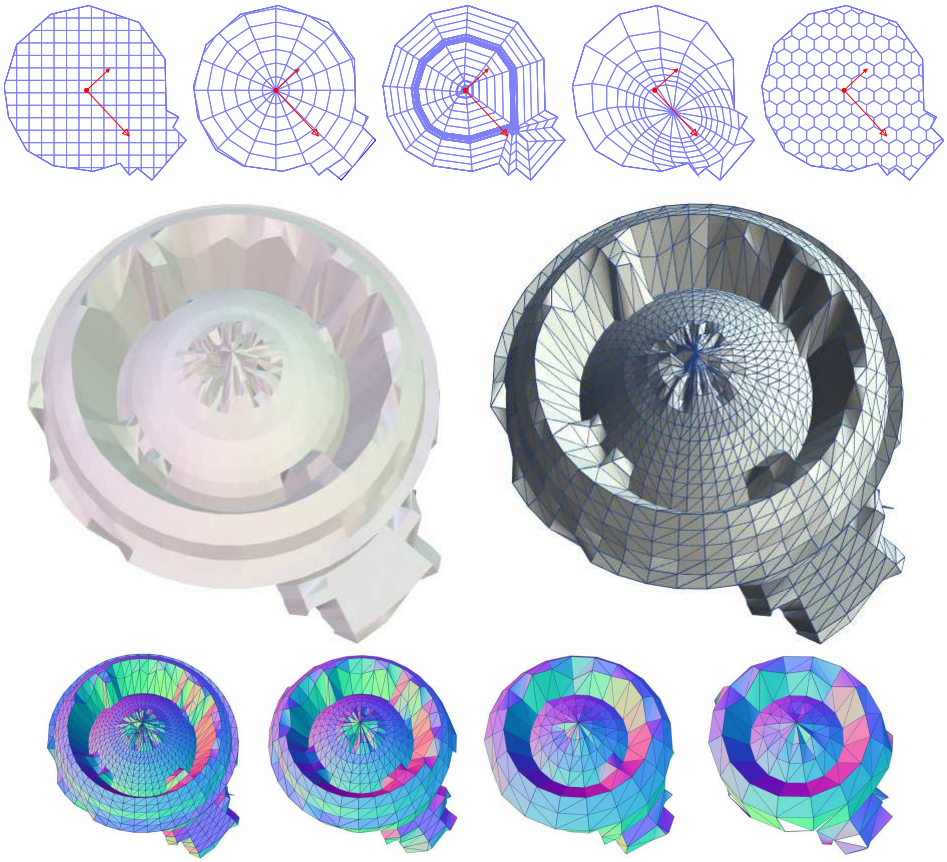

Figure 3.23: *structured polygon sub-division functions are exploited to control the distribution of vertices across roof-shapes during non-linear projection*

The core underlying aim of the sub-division functions is to address the problem of modelling non-linearity, without resorting to conformal-constrained [142] or general Delaunay triangulation (as in [12] and [71]). One of the key benefits is reducing the time to construct irregular roof-surface-meshes.

Figure 3.23 aims to clarify what is meant, by a structured sub-division, using the example of the Manchester library from §3.1. From left to right, top to base, it illustrates the *grid-split*, *radial-split*, *straight-line-skeleton-split*, *convergence-split* and *hexagonal-split* sub-division functions. Then the result of projecting the error-minimising radial split division and its topology. The last row illustrates the effect of varying the roof error tolerance.

As polygonal clipping is generally a real-time process [124], the sub-division projection executes orders of magnitude faster than an equivalent triangulation. However beyond simply the computational efficiency of the approach the critical benefit (and advantage relative to the use of a DT or CCDT) is explicit control of the topology of irregular roof-surfaces.

These explicit projection methods (linear and non-linear) provide a robust (accurate) means to position each 2D shape in 3D space. However one of the problems you'll notice (especially in the planar projection illustrated in figure 3.21) is that discontinuities can occur in the elevation at shared edges between neighbouring roof-shapes. This is because although the set of 2D vector shapes is non-self-intersecting (free of cracks and overlap) in 2D, the resultant projected 3D vector shapes may still be subject to height discontinuities as a result of non-uniform sensing noise. [134]. The problem is apparent in the linear-projections and is also characteristic of the models typically returned by data-driven plane based reconstructive methods. [95]. The MAMMAL algorithm resolves these issues by exploiting the shape-graph computed during vectorisation in order to optimise height-discontinuities between neighbouring roof-shapes.

Graph-Based Topology Refinement in 3D

The aim of graph-based topology refinement in 3D is to enhance the visual quality of the linear (planar) projected building models by refactoring roof-patch surface meshes in order to remove discontinuities in elevation between neighbouring roof-shapes. Although this stage is not strictly necessary in the case of surveying and exploiting the projected models for analytic tasks, it is incredibly useful in instances where the projected models are to be exploited for visualisation and simulation. Essentially this will not improve the geometric error associated with each result, but rather seeks to improve the aesthetic attributes of each reconstructed model.

Using the graph of roof-shapes, refining each projected model is handled by the MAMMAL algorithm with two simple graph-transformations:

- 1) **Clustering Adjacent Graph Nodes (Vertices)** based on deviance in elevation - such that multiple nodes that are close to one another converge into single nodes. This is analogous to a vertex snapping procedure in the sense that multiple vertices produce single vertices.
- 2) **Collapsing Redundant Graph Loops (Faces)** based on whether they become degenerate as a result of the preceding clustering - such that the undesirable roof-shape elements are removed. This relies on the observation that generally roof-shapes that collapse to lines, points or self-intersecting polygons, are generally insignificant. However for this loop-collapse to work effectively it is imperative that the input roof-shape-net does not contain any degeneracies to begin with.

Figure 3.24 illustrates the graph-refinement in 3D using a building from the 25cm point-spacing city of Manchester dataset. Interestingly although the primary reason for this post-process is to enhance the visual appearance of the models, an un-expected by-product is an overall reduction in the number of geometric primitives required to characterise each building - supporting the maximal-area minimal-primitives reconstructive principle.

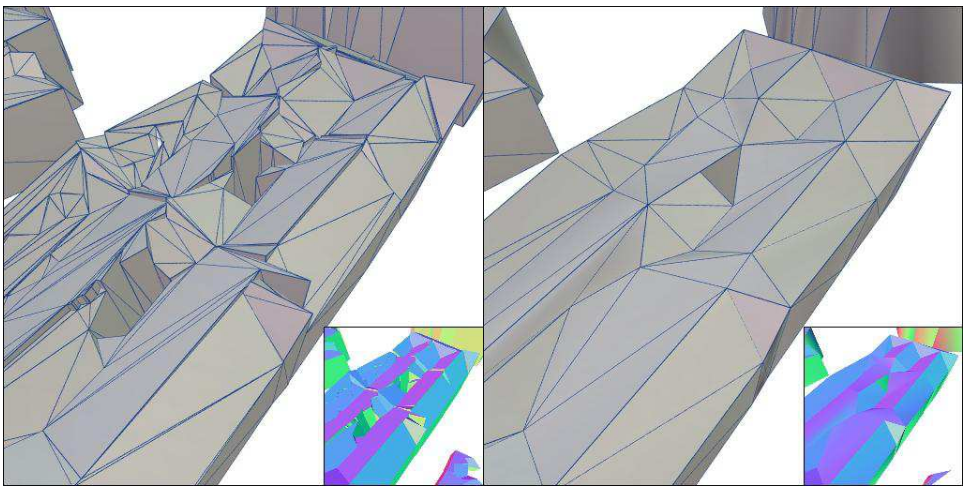

Figure 3.24: *graph-refinement turns a planar model with height-discontinuities into a higher-quality continuous model, illustrating (left) the unrefactored linear-projection using distinct, PCA plane components, and (right) the graph-refined result with a node-merging elevation threshold of 50cm*

The underlying insight is that by exploiting a graph representation of each building's roof-shapes, point-location and topological queries can be determined more efficiently than with an set of disjoint planar pieces. Beyond its efficiency the shape-graph provides a generic abstraction that can be traversed and transformed using a myriad of different operators. Whilst in this research two simple data-driven graph-transformations are used, future investigation might consider, higher-level graph transformations such as treating the problem as a generalised energy minimisation.

Further the reason that this is preferable to the quadratic error-minimiser projection (in the case of linear projections) is that it does not force continuity - i.e. it allows step-edges to be combined with continuous edges.

To revise - the projection process employed by the MAMMAL algorithm, computes a mapping between 2D roof-shape-nets and 3D surface-meshes, via a *parallax-pop-up*. Note though that whilst graph based topology refinement in 3D enhances model quality, the key to efficient projection lies in the calculation of geometric error. Basically in order for the MAMMAL algorithm to quickly traverse the space of possible projections a fast means of evaluating the geometric fit of each projected model is required. Vitally the rate at which MAMMAL can measure geometric error is also critical to the performance of the final stage - model optimisation. For this reason, (before discussing optimisation), this section details the efficient error measure exploited to compare a polygon-mesh to a target cluster of range-points.

Measuring Geometric Error in 3D

The MAMMAL algorithm exploits depth-buffer rasterisation in order to create regularly spaced displacements maps with equivalent point-spacing to the input range-scan during the evaluation of each procedurally generated model. During experimentation point-to-plane [156], and ray-casting [124], error functions were also tested as alternative methods of creating the elevation grids necessary for quan-

titative analysis. As a spoiler - the core reason for exploiting a rasterized depth-buffer over point-to-plane or ray-casting is computational efficiency. Although this is discussed in more detail shortly, for now simply appreciate that rasterisation executes 2-3 orders of magnitude faster that than an analogous depth-buffer construction using ray-casting or point-to-plane distances - yet returns a result with negligible variance relative to the other two methods. Which means the same comparative operation can be performed in less time. There are a couple of downsides (/potential limitations) of this approach which are also discussed.

Formally calculation of each model's RMS error in three-dimensions is:

$$rms(P,C) \leftarrow \sqrt{\mu(\sum_{i=0}^{N}(D_{Chebyshev}(P_i,C_i))^2)} \tag{3.3}$$

where: P is the cluster of range points, C is the discretised candidate model, $|P| = |C| = N$ and $\forall p \in P : \exists\ c \in C\ s.t. |p-c| = \sqrt{\sum(p_i - c_i)^2} = max(|p_0 - c_0|, |p_1 - c_1|, |p_2 - c_2|) = \sum |p_i - c_i|$. This expressions states that the error between a target elevation grid (P) and a discretised models elevation grid (C) is the square-root of the mean of each corresponding point's elevation difference squared. In essence at each position, the magnitude of the variance between the model's surface and the input range scan is considered. Furthermore, because the target and candidate elevation maps are aligned to the same coordinate system, this resolves to the Chebyshev distance. Another way to think about this is the RMS of a difference matrix (D) that results from $P - C$. The key insight is not this error formulation in and of itself, but the observation that due to the 2.5D (parallax) nature of the input, an object order (as opposed to a pixel/point-order) algorithm can be used to construct the matrix C. Since the matrix P is constant (the target range points) by minimising the time taken to construct C, significant performance enhancements are achieved. Whilst it is incredibly simple, this is key to fast 2D to 3D projection and the parametric optimisation.

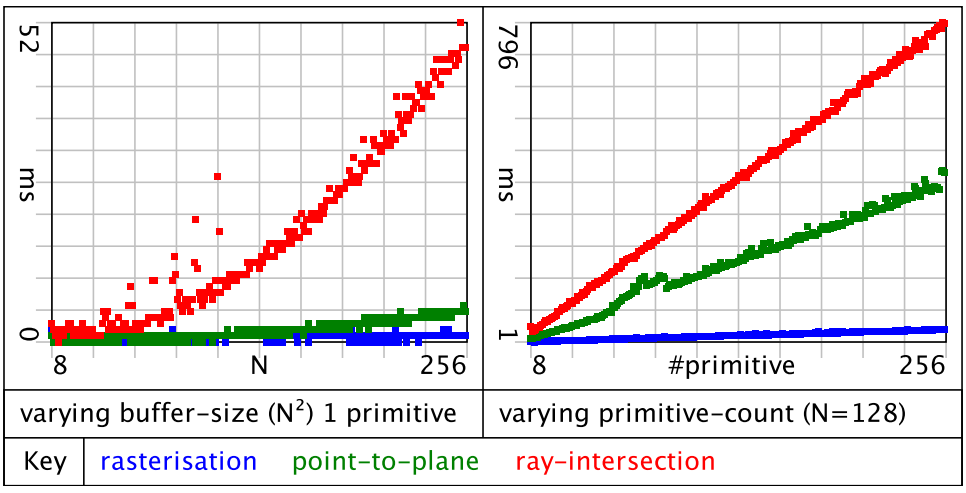

Figure 3.25: *profiling the growth in runtime for point-to-plane, ray-intersection and rasterisation depth-buffer construction for the parallax-error-functions: the left graph plots the variance in runtime as the size of the constructed depth buffer increases (NxN) - using a single primitive, whilst the right graph plots the runtime using a constant depth-buffer size*

(128x128) and a variable number of primitives: the results were generated with compiler optimisations turned off and spatial-optimisations (KD-tree, oct-tree, bucketing) disabled

Figure 3.25 plots the key factor in the algorithms efficiency. It demonstrates the significance of the variance in execution time for depth-buffer construction using the methods described. In particular, computing geometric error between a sparse model and a cluster of range-points is such a vital component of an automatic technique - that minimising the time it takes, can (and is shown to) significantly enhance performance. Fast calculation of parallax error, drives the MAMMAL algorithm. Essentially without the rasterised depth-buffer error, execution time degrades significantly. The key logic is that by discretising candidate building models using rasterisation, the MAMMAL algorithm can difference range-points and polygonal-meshes by taking the difference of two scalar matrices (and omitting square-roots).

As stated, both point-to-plane and ray-intersection error measures could also be used to create equivalent discretisations. However the thing to understand is that rasterisation is a real-time process. Millions of triangles can be rasterised in milliseconds, where as ray-intersection exhibits exponential run-time growth. This is because by rasterising, for each point in the depth-buffer the algorithm can calculate displacement without explicitly determining which triangle is closest to the point. This is also the key difference between object-order and image-order rendering algorithms. This is evidenced by the steep growth in execution time for ray-casting (in-particular) in figure 3.25 as the number of primitives increases, whilst the runtime associated with rasterisation grows very-slowly and remains negligible relative to the other two methods. Further this is only possible because of the 2.5D nature of aerial-laser scans. Ultimately fast calculation of geometric-error is also vital to the performance of the parametric optimisation.

Note: although this is not the first algorithm to exploit a rasterised depth buffer in calculating error, it is (to the authors' knowledge) the first instance of its use to drive non-linear functionally-based geometric optimisation of 2.5D architectural models. In this sense, the exploitation of a rasterised depth buffer, places the MAMMAL algorithm in line with image based optimisation operators, because rather than performing multi-modal optimisation (points relative to triangulated surfaces which is most common in geometric algorithms such as [129], [118], [157], [143] and [75]), MAMMAL's strategy is driven by image-like discrete rectilinear grids.

Before progressing to the final stage (optimisation) this section briefly considers the down-sides of the prescribed error-measure. There are two limitations of using this depth-buffer error measure. The first is shared by all the parallax functions, the second is specific to the rasterisation process.

- **Unsuitable for unstructured ground based scans** - since this relies on perpendicular displacement, the depth-buffer error will fail in the case of point-clouds that do not obey the 2.5D principle. This additionally applies to airborne scans that capture subtle non-parallax facade details (such as those resulting from low-altitude UAV scanning systems). Essentially this method of fast error calculation is designed for parallax geometric representations only - the downside of this being that the efficiency gained comes at the cost of generalisation.
- **Difficulty discretising skinny (sliver or near degenerate) triangles** - as a product of the manner in which input triangular facets are mapped to discrete

grid locations. This is a prevalent problem for rasterisation algorithms in general. The most common solution is to super-sample the elevation grid and/or employ floating-point-arithmetic over integer-arithmetic. However more robust strategies also consider the adjacency of triangular edges in the original mesh in order to prevent any such gaps that may otherwise still result.

This section has explained the process of generating 2.5D projected surface models from sets of vectorised 2D shapes. It covered the two types of projection (supported by the MAMMAL algorithm) linear and non-linear, as well as the 3D-version of the graph-refinement routine (exploited to resolve discontinuities in elevation between neighbouring roof-shape edges). Additionally it introduced a fast rasterised depth buffer error-measure which is crucial to minimising runtime and outlined the limitations that an implementer should be aware of. At this stage the MAMMAL algorithm has recovered accurate sparse geometric models of the buildings in the input airborne range scan. Up to this point, the algorithmic modelling logic has been entirely data-driven, in the sense that the output of each sub-stage is controlled by the input points or vector shapes. Whilst this behaviour is advantageous in many respects (refer to this chapters overview for revision), it also means that MAMMAL is sensitive to low quality input point-sets. This includes point-clouds of low-resolution (high-point-spacing), point-clouds with partial or missing data and regions affected by non-uniform sensing noise and low-quality pre-processory filtering. The good thing is that this flip-side largely only affects the process of roof-shape projection (which turns the 2D shapes into 3D models). Generally speaking, the segmentation and vectorisation stages are less prone to this (due in part to the stability of the difference of elevations models and the maximal area roof-shape segmentation). However one of the key problems identified in examination of the pre-existing methods is/was the rate at which the integrity of the resultant geometric models degrades with low-quality point-input. In an ideal world, a technician would simply resolve to ensure beautifully sampled point-clouds were supplied as input. However often this is not always possible, since the economic cost of re-scanning may be prohibitive. The problem is compounded by the fact that frequently the artefacts that degrade a generated model's quality, occur sporadically in datasets - which makes their uniform estimation and resolution harder. The simple fact is that, some classes of material cannot always be scanned perfectly. Typically this is a result of the irregularity of their reflectance properties (as is the case for glass), but further the spatial scale of the features present upon a surface can also have an undesirable impact (such as in the case of undulating or corrugated surfaces). For this reason, model-optimisation strategies are employed. They may be data-driven or model-driven, however the unifying goal is to improve the structural quality of building models recovered from low quality clusters of points.

3.3.5 Optimisation

The driving aim of model optimisation is to enhance the visual and structural quality of projected building masses recovered from low-resolution scans, such as the 1m point-spacing Bath dataset. Optimisation also enhances models recovered from higher resolution scans, by mitigating additional sensing artefacts. Geometric optimisation is a highly active field of study, with many pre-existing works focussed on generalised energy minimisation (since many geometric problems such as mesh parameterisation and deformation can be reformulated as either minimisation or maximisation tasks).

Whilst such strategies are excellent in terms of their generality (and their ability to be reused) the key limitation is the high convergence times that result as the complexity of the optimisation problem grows. Often such algorithms can take minutes to converge for a single geometric representation. Although this may be acceptable for processing individual *hero-models*, in general such performance is insufficient for large urban scenes consisting of a multitude of objects. In essence such methods are good for abstract problems because they allow one to define the objective in terms of an energy function $f(x)$ and a set of (typically linear) constraints, such as: s.t. $x \geq 0$. However this level of generality degrades their execution time.

Alternatively some researchers [1], [39], [108] exploit random sampling strategies, with methods based on the RANSAC paradigm dominating. The goal of the stochastic optimisation techniques is to iteratively localise on accurate compact approximations in an efficient manner, by generating candidates randomly (so as to more efficiently traverse the solution space).

Unfortunately early experiments demonstrated that neither approach is perfect for this class of optimisation problem. The numerical approaches (such as Newton's method, gradient-descent and accelerated/boosted gradient-descent) are quite-heavy weight, and require a convex solution space since non-convex problems can result in convergence on a local minima. The stochastic methods on the other hand offer few guarantees of convergence and require more involved evaluative measures since (even with seeded random generators) their performance is considered probabilistically. Ultimately however the ideal scenario is a model optimisation approach that amalgamates the positive features of the numerical and stochastic approaches whilst negating their undesirable aspects. Essentially we seek a numerically robust deterministic optimisation method that converges efficiently.

This portion of the chapter introduces a novel procedurally based optimisation strategy designed specifically for 2.5D parallax building mass-models. The approach relies upon a number of observations about the nature of the domain, in order to enable clean, compact, semantically rich geometric models to be extracted efficiently. The key concepts introduced are:

- **Geometric Model Selection Criteria** - as an abstract means to control the algorithms decision-making in ranking optimisation candidates.
- **Fixed-Form Parametric Model Detectors** - as a means to define and search common and regular generative modelling functions.
- **Open Parametric Model Detectors** - as a means to define and search specialised and irregular generative modelling functions.

- **Constructive Solid Remassing** - as the method of unifying a set of distinct parametric masses to yield a watertight building shell.
- **Automatic Interface Generation** - as an advantageous feature of the proposed procedural optimisation strategy and in-particular the dual nature of the implicit representations that are exploited.
- **General-Purpose Procedural (Programmable) Optimisation Kernel** - as a generalisation of this approach to parallax model optimisation and the factors necessary to enable additional user-written functions to be supplied post compilation - (support for expansion).

At a high-level the optimisation stage behaves in the following manner. For each building, it constructs a set of *candidate* mass-models by varying the input arguments to a finite number of generative modelling functions. It then ranks the candidates models using an abstract selection criteria and returns the highest ranked candidate mass-model. Of the functions used to generate candidate models, there are two basic types: fixed-form and open parametric modelling functions. The differences between these two types of shape detector are discussed in greater detail shortly. The key aspect is that they provide complementary mechanisms for framing model-based and data-driven approximation functions. The next sections cover the key conceptual components of the proposed non-linear building optimisation routine.

Parametric Optimisation of Low Quality Models

The approach described, aims most of all to conform to two key desires. It should converge quickly and yield consistent (repeatable) results. Given that no single optimisation strategy can be considered optimal for all classes of building, the approach instead exploits a number of different parametric modelling functions. Although this ideology shares similarities with the generalised approach of Lafarge, Mallet et al. [72], [73] - the underlying reasoning is distinct. This strategy exploits the observation that human CAD technicians typically have a number of distinct modelling strategies in their arsenal, and opt for one over another based on the requirements to model a specific instance of an object. In a similar manner, the MAMMAL algorithm's approach embodies a variety of distinct common building modelling approaches. The critical difference is that whilst a human can efficiently alter a mass-model directly (by moving vertices and faces to better represent the target object), the same approach applied algorithmically exhibits poor runtime performance since the number of elements (vertices, faces) in a mesh can grow quickly. Hence parametric modelling functions are exploited instead because they embody semantic meaning, and they dramatically reduce the complexity (order) of the optimisation task down to 1-N arguments - as a product of the additional layer of abstraction. The core idea, is to exploit dynamic descriptors that may be efficiently manipulated in order to identify characteristic approximations of a target object - rather than lower level primitive (vertex, edge, faces) refactoring strategies.

Another critical difference, is that unlike a human CAD technician - the MAMMAL algorithm cannot say ahead-of-time which approach (parametric function) is best suited to a cluster of points. Although one could try to frame pre-processory filtering functions that aim to embody a humans perception of the mapping between a cluster of points and an appropriate class of parametric model, in practice this is a

cumbersome process. Rather the MAMMAL algorithm must consider all possible modelling functions.

Basically a human CAD technician, can look at a picture or point-set of a roof and determine instantly whether it represents a Gable or Mansard building (for example). However the MAMMAL algorithm possesses no such perceptual prowess - and as such can only state the class of function suitable for approximating a cluster of points after it has considered all possible classes. This is both good and bad. Good in the sense that it is exhaustive (and so will never skip or miss out on good parameterisations), but bad because it means additional compute-power is used. Later on a number of optimisations (to the optimisation process) are discussed that seek to provide early *get-out* clauses in order to reduce the overall optimisation time.

Essentially optimisation iterates over a finite number of fixed-form and open generative modelling functions, and adds instances to a candidate set. Then it sorts (ranks) the candidate set using a selection criteria that specifies the objective function and returns the highest ranked optimisation candidate.

The final (optional) water-tighting stage exploits a simple optimised version of the binary space partitioning algorithm for constructive boolean operation on polyhedral mesh. However before covering each of these optimisation stages, the pro and cons of this procedural approach are explained.

The advantages of this approach include:

- Separation of Geometric Descriptor from Numerical Optimisation : for which the key benefits are Generality, Simplicity and Expandability.
- Support for Non-Convex Optimisation : such that a Local Minima of a Non-Linear Solution Space can be deterministically resolved.
- Support for Embedding Conditional Statements : algorithmic flow-control logic, such as branching *if* and *switch* statements in order to define heuristic constraints on the geometric candidates generated.
- Dynamic Functional Descriptors : enable a multitude of distinct model instances to be encoded in a single generative modelling routine.
- Duality of the Geometric Representation : which means the same descriptor is both suitable for Forward (User-Centric) Parametric Modelling and conversely Backward (Data-Driven) Model Reconstruction
- Negation of associated IO overhead: since each model is defined implicitly there is no IO associated with their construction or evaluation : this additionally reduces memory limitations because candidates can be destroyed and recreated as necessary to control resource allocation.

These benefits should provide some insight into the rationale underlying this investigative path. Nonetheless there are a number of limitations that should also be noted. The disadvantages of this approach include:

- Requires Highly-Efficient Error Measure : this method of model optimisation is only feasible if the time taken to generate and evaluate a procedural model (relative to a target cluster of points) is negligible.

- Non-Trivial to Generalise to Unstructured 3D Scans : because although candidate model generation performs similarly in 3D to in 2.5D, the evaluation stage exhibits far greater computational expense. Fundamentally an error-measure of equivalent efficiency to the rasterised-depth-buffer is required in 3D. Whilst an approximate (projection-based) error measure could be used, one must note that the efficacy of the optimisation drops significantly. The step from 2.5D to 3D requires a point-to-plane error-routine which cannot easily be replaced for a faster error-measure without the loss of accuracy or generality.

The next subsections explain the concepts outlined in greater detail.

Geometric-Selection-Criteria

The process of parameterising a cluster of points produces a set of candidate polyhedra mesh from which the algorithm must decide which is most suitable. For this the notion of selection criteria is used. Each selection criteria specifies the key property of the candidate mesh that the algorithm should return given a finite set of options for approximating a point-cluster. For example if the minimum-vertex-count selector is supplied as input, the algorithm determines all candidate parametric meshes with geometric error under the user supplied tolerance, and returns (from this subset) the model with the smallest number of vertices. Formally these criteria are defined as:

- **minimum_mean_point_error** → the candidate that is the global point error minimiser - induces the least point-to-plane elevation error

$$min : \sum abs(p_i - c_i)/|P|$$

- **minimum_volumetric_error** → the candidate that is the global volumetric error minimiser - induces the least volumetric variance

$$min : abs(\nu(P) - \nu(C))$$

- **minimum_surface_error** → the candidate that is the global surface error minimiser - induces the least change in the boundary meshes surface area - relative to an interpolation of the original (target) points

$$min : abs(\iint_S (P) - \iint_S (C))$$

- **minimum_vertex_count** → the candidate that uses the fewest vertices to meet the user-supplied error tolerance - induces the sparsest under error - with a preference for polyhedra with shared vertices

$$min : |C_{vertices}| \; \forall \; C \in CS \; s.t. \; error(C, P) < \alpha$$

- **minimum_face_count** → the candidate that uses the fewest polygon faces to meet the user-supplied error tolerance - induces the sparsest under error - taking into account the topology of the mesh that will be written to disk : without punishing the presence of duplicate vertices

$$min : |C_{indices}| \; \forall \; C \in CS \; s.t. \; error(C, P) < \alpha$$

- **maximum_quality** → the candidate with the highest mesh quality that meets the user-supplied error tolerance - derived from a weighted sum of the ratios of each faces internal angles and surface-area

$$max : Q(C_{vertices}, C_{indices})$$

The first three criteria address the requirement for geometric-accuracy, whilst the remaining three deal specifically with the quality and sparsity (level of compression) of the optimised models. Vitally these criteria provide a means to automatically resolve different classes of parametric model, that is both controllable by an end-user and abstracted from the domain. They enable the MAMMAL algorithm to reason about the idealised return without the manual specification of discrete levels-of-detail (as in [156]).

Essentially the selection criteria allow an end-user to specify at a high-level the types of model they desire in a uniform and transparent way. The MAMMAL algorithm then does the grunt work creating potential optimised models and determines and returns the *best* one, based on the supplied criteria. Good analogies for the selection criteria include as a *reconstructive-goal* and as *ranking* or *objective* functions. However readers familiar with CGAL and similar APIs should recognise this as the simple use of predicates to define a geometric goal and guide the traversal of an abstract space.

These intuitive selection criteria allow the MAMMAL algorithm to automatically decide how best to approximate a building during optimisation - given a number of candidates. Based on this, the next sections discuss the generative functions that return candidate models for each cluster of points.

Candidate Generation and Verification

The key to the MAMMAL algorithm's efficiency is the manner of generating and differencing parametric models with the original range points.

As stated the MAMMAL algorithm considers two-types of generative function. Fixed-form (templated) and open (data-driven) modelling functions.

Fixed-form-functions behave as typical parametric models - taking in a finite number of arguments that control the geometry they return. They are effectively dynamic 'model-library' [124] functions. The MAMMAL algorithm traverses each fixed-form function's parameter space using discrete steps - in order to identify values for the input arguments that minimise the geometric error of the resulting model relative to each target cluster of points. Open-functions on the other hand, implement data-driven modelling strategies - whose return value is directly controlled by the input building cluster.

The open-functions provide greater flexibility in characterising irregular classes of building. However the flexibility to define non-linear data-driven geometric-detectors, comes at the cost of efficiency. The vital difference between the two types of function is that fixed-form functions are un-aware of the input scan data, whilst open functions derive their geometric return directly from a cluster of segmented points. This distinction is very similar to the fixed-form vs open 2D shape detectors exploited during vectorisation. The core idea is that MAMMAL has two complementary types of optimisation function - that are suited to prior based fitting

and data-driven approximation.

Note: that these two approaches to candidate generation, enable the MAMMAL algorithm to more efficiently traverse the solution space relative to rigid-body, affine and even deformable transformation models. Interestingly though, they both conform to the same verification process. To evaluate a candidate model, a discretised version is created using the rasterised depth-buffer explained in $3.3.3, and compared to the original (target) DEM points. This reuse of the fast parallax error measure, results in the verification of parametric models being semantically no different to that of the previously recovered projected models. Later discussions present enhancements to the optimisation stage. Next, the fixed-form and open detectors are explained.

Fixed-Form Shape Detectors

The MAMMAL algorithm exploits a set of simple generative functions to characterise common-architectural forms. An intuitive example is the TaperBox function. Figure 3.26 illustrates a sub-set of the space of potential models that the TaperBox function can represent. You'll notice hipped, Gable, Mansard, Gambrel, shed, pyramid-hipped and flat roof-types.

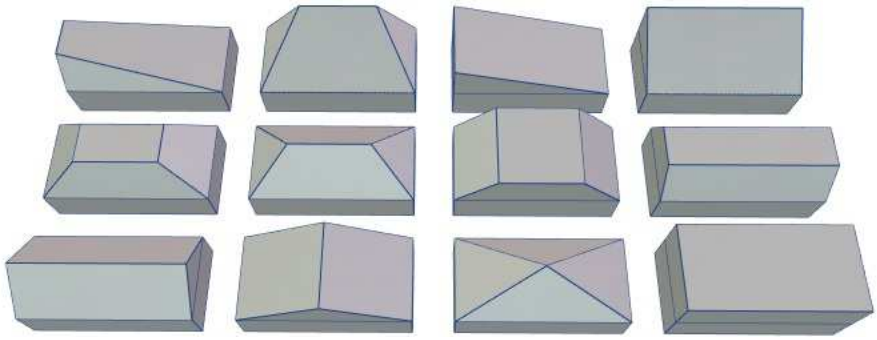

Figure 3.26: *examples of instances of 2.5D mass-models from the space of potential returns for the parametric fixed-form function -* ***TaperBox***

The key insight is that each of the instances illustrated in figure 3.26 is simply the product of varying four scalar parameters (input arguments) to the TaperBox function - scale_x, scale_z, offset_x and offset_z.

This example (in-particular) relies on the topological symmetry present in the TaperBox - but further it also demonstrates that highly expressive distinct roof-types can be expressed as the product of a very small (typically less than 8) number of non-linear transformations. Whilst the resulting function is intuitive to understand, in practice writing an analytic error-measure for such a parametric function quickly becomes intractable as the irregularity of the representation grows. Additionally it has the tendency to constrain (and/or impede) the definition of the architectural representation. As such the fast parallax error measure is vital - since it dissasociates the functional descriptor from the evaluation (error/energy minimisation) process.

Further, the dynamic nature of the representation (a model-generating parametric function) signifies a major benefit over pre-existing model-template based reconstructive methods [1], [71], [75], in that whilst a number of object descriptors

would be required to accomadate each distinct roof-type (illustrated in figure 3.26), the MAMMAL algorithm requires a single object descriptor to embody the same meaning. The implication of this lies in enabling interactive user-centric parametric manipulation and modification of the reconstructed buildings - which is discussed in greater depth shortly.

In order to better explain these ideas, two additional fixed-form functions are covered. The RadialRail and the TaperBoxGroup functions.

The RadialRail function (depicted in figure 3.27) is a simple circular footprint mass detector that exploits a set of distinct *revolved* profiles to approximate circle based building roof components. Each input profile can also be scaled non-uniformly to alter the dimensions of the mass returned, yet they are all a product of the same generative modelling function.

By this point, an astute reader should have begun to notice the similarities present in the fixed-form functions. Generally speaking they are all derived from simplexes and mapped to 3D using a low-level modelling procedure (a non-linear extrusion in the case of the TaperBox and a generalised cylinder in the case of the RadialRail). The reasoning behind classifying them as fixed-form is that ultimately each is *fixed* in the *forms* it can represent. This label does not mean the fixed-form functions are rigid or necessarily linear (in reality most are not), rather it states that the scope of the *dynamism* achievable is fixed ahead of time - by the author of the function.

Figure 3.27: *models returned by the function -* ***RadialRail***

One of the key benefits of library-based reconstructive methods is that they enable priors and constraints to be imposed so as to preserve parallelism, symmetry and other desirable geometric features. Beyond embedding heuristics about roof-shape arrangements, an additional benefit is that these functions are also exploited internally by the MAMMAL algorithm in the detection of repeated structures and common patterns prevalent on roofs.

For example by trivially adding an additional two parameters - repeat_x and repeat_z (that both represent integer *instance* counts) - to the TaperBox function, one enables repeated rectilinear structures to be characterised - such as the M-shaped and corrugated roofs illustrated in figure 3.28.

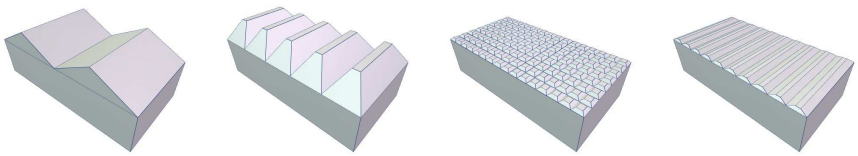

Figure 3.28: *models returned by function -* ***TaperBoxGroup***

To revise, this subsection covered the fundamentals of the fixed-form parametric detectors - including the aims and advantages and provided a handful of clarifying examples. The next-section covers the parameterisation of more complex topologies, using the example of the terrace-detector to help clarify.

Open Shape Detectors

Due (in large) to the efficiency of the depth-buffer error detector (and the fact the IO overhead for parameterisation is nil) the optimisation method employed by the MAMMAL algorithm can generate and evaluate hundreds of thousands of varying candidate models in a surprisingly efficient manner - without any heuristic guidance. However for certain classes of building (especially those with high-dimension corresponding parametric functions), such an exhaustive approach can significantly degrade performance.

The heart of the problem is that the operator is effectively trying to traverse the infinite space of potential geometries. Even with discrete steps, as the number of parameters grows the complexity of identifying an appropriate model also grows. To address this the MAMMAL algorithm generates candidates for open (complex) functions differently. Each open-function is responsible for instantiating candidate models for an input building cluster.

In this manner each open-function acts as a self-contained detector - in the sense that MAMMAL no longer controls the traversal of the solution space defined by the function, rather the function itself encompasses the required logic. So whilst each fixed-form function accepts a finite (though variable) number of input arguments and returns a single mass-model:

MassModel FunctionName(type1 arg1, type2 arg2,... typeN argN) { *...* }

each open function accepts a single input object descriptor and returns a set of candidates based on its internalised data-driven modelling logic:

MassModel[] FunctionName(BuildingDescriptor descriptor) { *...* }

MAMMAL then adds each element in the set of candidates to the global candidate-set exactly as it would for the return of a fixed-form function.

This subtle difference aims to both speed up the evaluation of highly-specialised detectors and enable a generative function to directly access the target cluster of points and associated vectorisation data during model creation.

Although this may be hard to imagine, what it means is that each open-shape detector can directly interrogate a target building object (segmented points, vector shapes and projected mass) in order to alter the nature of its return value so as to better characterise the object during optimisation.

One particularly intuitive open function is the terrace-detector. It is responsible for parameterising, *snake-like* [155] residential buildings.

The underlying ideology is similar to the method of extracting and refining transport profiles to describe radially symmetric architecture in facade scans by Wu et al. [151]. We drew further inspiration from the 2D-snake algorithm of Yan [155] et al., however unlike our predecessors the operator favours a direct geometric technique over energy minimisation.

To derive compact terrace models the TerraceRail function exploits the *graph-refined approximate interior linear spine* algorithm (GRAILS) introduced in §3.3.3, to first localise on an interior polyline representation of each cluster of points. The TerraceRail function then exploits this *spine* to *sweep* 2D slice profiles about the extents of the building to yield 2.5D mass-models.

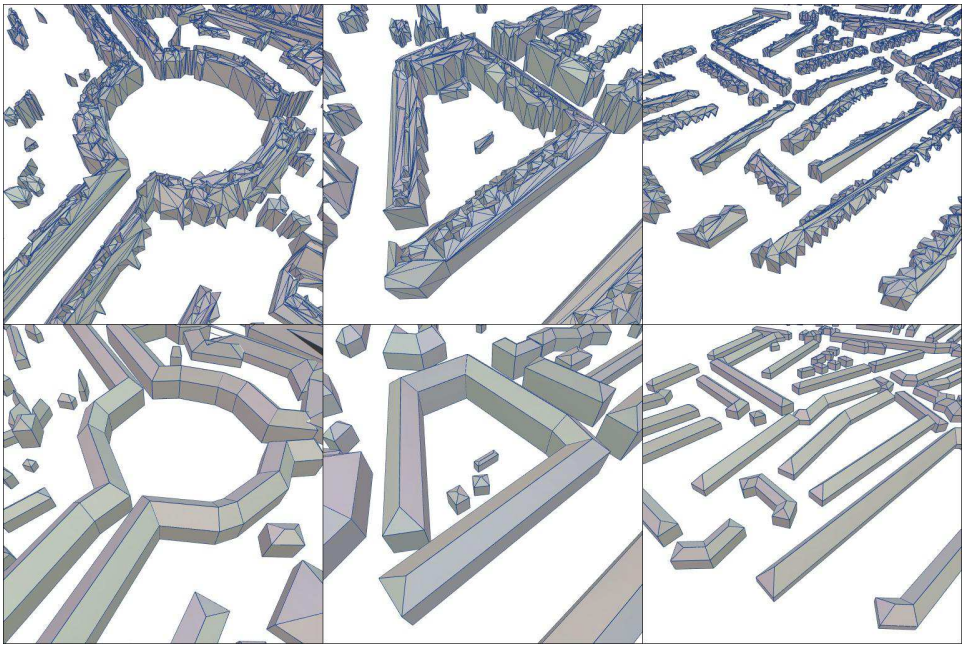

Figure 3.29: *terraced buildings at 1m resolution in the city of Bath dataset - the top row is the unrefined result whilst the second row shows the result using the data-driven terrace detection function* ***TerraceRail***

Vitally the terrace extractor is not constrained to monotonic polygons (see middle column of figure 3.20). Despite its heuristic nature it yields stable results even at low point-spacings. Figure 3.29 illustrates further.

This example open function seeks to demonstrate the benefit of having self-contained mass detectors capable of optimising irregular architectural forms. Ultimately though the true power of the fixed-form and open-functions is most apparent when applied not only to each individual building but to each segmented roof-shape in turn. Essentially if the MAMMAL algorithm only optimised point-clusters at the level of individual buildings, then a plethora of open-functions would be required to handle the diversity present in architectural roof-shape arrangements. However because MAMMAL has already isolated salient components of each building's roof (during segmentation), the optimisation stage can also be applied to each roof-shape in turn in exactly the same manner as for each individual building. This means that rather than having to evaluate a large number of parametric functions (to optimise complex roofs), a comparatively small number of simple functions can be composited to yield a constructive building representation.

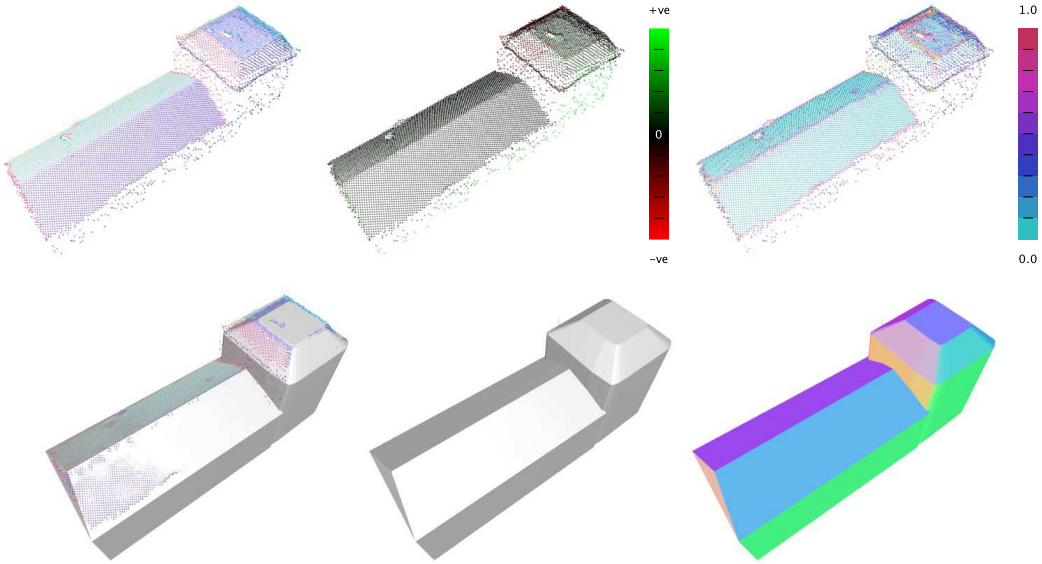

Figure 3.30: *an example of the parameterisation of a simple compound building from the 25cm point-spacing Manchester dataset: depicting (from left to right, top to base), the input building points coloured by normal, the point-to-model elevation error (black maps to zero, green to positive error, red to negative error), the normal variance between normals of the input points and parametric model (using a HSB colour scheme such that hue 0.5 maps to 0), output parameterisation overlain with input points, output parameterisation rendered in isolation and output rendered with normal-to-colour shader.*

Figure 3.30 evinces this point using the simplest type of compound mass - a two component building. The MAMMAL algorithm relies on the ability to compose simple masses in a data-driven manner in order to form seemingly complex architectural mass-models as the product of a surprisingly small set of incredibly simple parametric functions. The final stage of the optimisation routine merges sets of parametric masses and is discussed next.

Constructive Solid Re-Massing

The final stage of parameterisation aims to unify the parametric masses recovered via optimisation in order to ensure the result for each building is a single manifold polyhedra mesh. Although this stage is optional the benefit of merging each parametric component is apparent when each building is used to guide the segmentation and reconstruction of unstructured ground scans. In such instances, the presence of non-manifold internal faces can degrade performance by introducing erroneous facades. Essentially this stage results in a single building shell from a set of manifold parametric masses.

An extended binary-space-partitioning algorithm is used to implement the constructive operations, from which the union is used to merge the parametric geometries. This stage effectively acts as a retopology operator in the sense that it preserves the extents of the object's geometric form, whilst altering the arrangement of polygonal faces used to characterise the extremal surface. Figure 3.31 illustrates this using a simple 3-component mass.

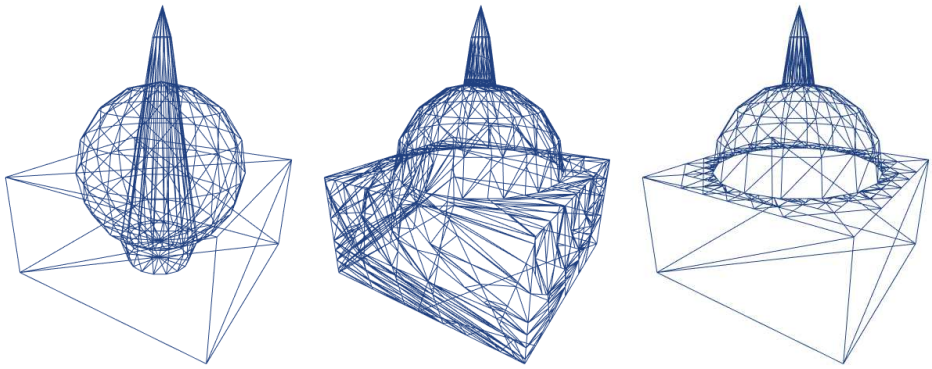

Figure 3.31: *a spatially aware variant of the binary-space partitioning (BSP) algorithm ensures a watertight-shell is returned for sets of parametric masses*

The left image illustrates the set of input parametric primitives (168 vertices, 332 faces), in the middle the result of the union operation for the set using the BSP algorithm (913 vertices, 1655 faces), and on the right the refactored return as a result of detecting and maintaining non-interacting planar pieces (500 vertices, 722 faces). The basic idea is incredibly simple.

The key insight is that in performing the boolean union operation on two polyhedra (A and B) only polygons that interact with the alternative operand actually need to be altered. This underlying idea was derived from observation of the limitations of half-plane clipping based methods. This is quite clear in figure 3.31. You'll notice that the vertical faces of the base cuboid are spatially distinct from the dome or round cone. Essentially if a polygon in representation A is unambiguously disjoint from the faces in representation B, then logically there is no reason to alter (split/-clip) the polygon in the output representation. As such the implementation boils down to determining the faces in a A that do not interact with B, the faces in B that do not interact with A, and clipping the remaining parts of A and B in the output.

Whilst this idea is incredibly simple to understand, in practice a naive implementation will generally lead to parity problems especially when coplanar primitives exist in the inputs A and B. The half-plane based boolean methods generally deal quite well with this by *double-clipping* (regularisation).

The limitation of this basic variant is that it relies on disjointness - which means that only faces that do not interact in any way with the other operand will be preserved. However often there are instances where most (or all - in the case of a co-linear edge) of a face lies outside of the other operand and should be preserved in the output. To support this desirable behaviour triangle and quadrilateral sub-division routines are exploited in order to break down each such face into sub faces so as to isolate as great an area of independent components as possible. The core concept is that by actively controlling the division of polygon faces prior to the boolean operation the variant not only manages the topology, but minimises the area of the clipped faces in the output mesh. This behaviour is evident in the top horizontal face of the cuboid in figure 3.31. Note that the corners of the refactored face (right) are composed of higher quality (larger surface area and smaller variance in internal angles) triangular facets than in the raw BSP result (middle). In particular this sub-divide enhancement is tantamount to a strategy for maximising the surface-

area of non-interacting polyhedra components between the input operands A and B prior to executing the boolean union.

In summary, the final (optional) process of the MAMMAL's algorithm's optimisation strategy is constructive solid re-massing, and involves merging and refactoring sets of parametric polyhedral mass-models in order to yield a watertight shell for each optimised reconstructed building-model.

3.3.6 Summary

The MAMMAL algorithm automatically recovers compact, 2.5D geometric models of buildings in airborne laser-scans using a four stage process. MAMMAL first segments the input data, using the difference of elevation models and maximal area roof-shape segmentation, in order to identify salient clusters of points. MAMMAL then vectorises the segmented points in 2D in order to yield 2D vector shapes representing the extremal boundaries of the segmented points. MAMMAL then projects the vectorised shapes into 3D by minimising the RMS error between each cluster of points and the polygonal faces of each projected model. MAMMAL then optimises each building parametrically in order to enhance the visual appearance and structural quality of models reconstructed from low-quality clusters of points.

Having explained the methodology, the next portion of this chapter presents the results of laboratory experiments in which the MAMMAL algorithm is exploited in order to reconstruct various city-scale datasets.

3.4 Experimental Results

This section of the chapter explains the evaluative measures that are used to quantify MAMMAL's performance and documents the results of executing MAMMAL on testing datasets. The primary aim is to clarify the physical performance of the MAMMAL algorithm when applied to real-world data.

Throughout this section the three primary attributes profiled are: geometric accuracy, level of compression and computational efficiency.

The aim is to provide a comprehensive account of MAMMAL's performance.

For each of the datasets, the additional discussions outline the pertinent aspects of the results and draw a readers attention to the key behavioural factors. However deeper explanations and the bulk of the critical examination is considered in the following portion of this chapter. As in the preceding chapter, here the focus is to enumerate the results and explain their meaning. The section that follows (analysis and evaluation) deals with exposition of their significance.

3.4.1 Synthetic Datasets

This section briefly outlines the results of controlled tests of the MAMMAL algorithm on synthetic data. The primary of aim of these experiments was to determine the behaviour of MAMMAL in controlled situations for which the expected reconstruction result was unambiguously known ahead of time. In terms of the methodology employed: the synthetic datasets were procedurally generated from manually modelled top-down building mass-models. This simply involved discretising polygon mesh at various sample spacings and introducing synthetic sensing noise (to varying extents) by jittering the elevation of points in the generated depth maps. To simulate the presence of missing data - a variable subset (a fraction) of the gridded points were removed (i.e. values set to the null-data indicator -9999).

The main findings from the controlled experiments are summarised following.

- Accuracy is bound by and tightly coupled to cell-size (resolution). This in particular applies to the data-driven projections. Essentially the greater the number of points per building cluster the tighter the fit of the generated models. However as scan resolution increases - so to does the relative cost of conforming to the MAMP principle (i.e. more post-processing is applied by the MARS algorithm to form maximal-area clusters).
- Synthetic results are less indicative of performance on real-data. Notably the performance on synthesised scans can mislead one as to the performance of an algorithm on actual scans exhibiting real sensing artefacts. This goes beyond simply high and low frequency noise. For example with the artificial tests problems such as terrain-estimation errors are non-existent and as such result in better segmentation results than occur in practice. Further real scans possess multi-scale features that perturb roof points and hinder segmentation whilst the synthetic scans capture fewer such features.

So whilst profiling on validatory synthetic data is useful from the perspective of understanding the limits of the MAMMAL algorithm (and indeed in development and debugging) it is not necessarily indicative of performance on real scan data.

3.4.2 City of Bath - 1ppm (1m)

The results of reconstructing the City of Bath are structured as followed. First a preliminary explanation of the experimental setup is provided. Following this, the results of segmentation, vectorisation, projection and optimisation are presented in turn. For each stage quantitative measures are provided in the tables and graphs whilst qualitative figures accompany the numeric results for further clarification.

Experimental Setup

The city of Bath experiments exploit the same airborne digital surface and terrain elevations models employed in the previous chapter's Semantic Change Detector. The DSM and DTM cover a 2km x 2.5km region of the south-west UK city, with a point-spacing of 1m. It represents the lower-resolution end of currently available off-the-shelf airborne scan data. Although in practice, the Bath dataset's point-spacing is too low for the reconstructed models to be exploited in the production of verified views or AVR's (accurate visual representations), they are still suitable for large scale rendering and visualisation. An additional benefit is the existence of a corresponding manually-constructed CAD model for qualitative inspection.

Segmentation Results

These results document the performance of MAMMAL's building detection (using the DoEM method described in the previous chapter) and individual roof-shape detection (using MARS). The key attributes profiled are the growth in execution time as a product of varied input arguments and the geometric characteristics of the set of salient regions identified by the stage. These results expose how effective the Maximal-Area Minimal Primitives principle is as a high-level segmentation strategy when low resolution point-data is supplied as input. They also demonstrate instances where the concept breaks down and the general limitations of the method.

DoEM: The following figures illustrate the outcome of DoEM segmentation.

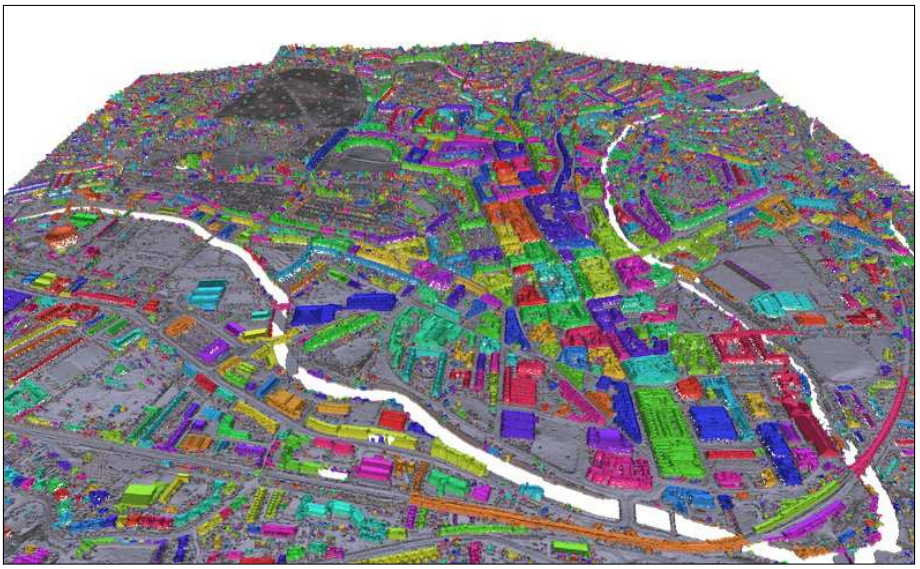

Figure 3.32: *buildings automatically identified in the city of Bath dataset*

Figure 3.32 illustrates the whole of the Bath dataset whilst figure 3.33 illustrates a

close-up view of a central subset of buildings. In both figures each building (connected component) is assigned a psuedo-random colour whilst the terrain and clutter is denoted by grey regions.

Figure 3.33: *close-up results of the difference of elevation models segmentation applied in order to identify buildings in the city of Bath dataset*

The qualitative results figures 3.32 and 3.33 indicate that despite the heuristic nature of the difference of elevation models, it results in stable identification of the individual buildings in the low resolution 1m Bath dataset.

MARS: The following figures illustrate the outcome of MARS on the city of Bath.

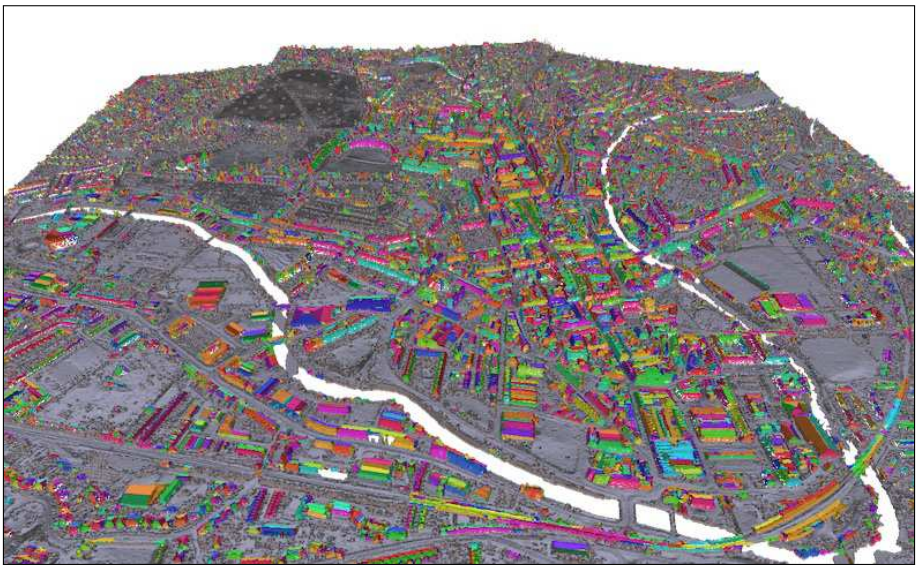

Figure 3.34: *roof-shapes automatically identified in the city of Bath dataset*

As in the DoEM figures, the ground terrain and clutter is indicated by gray regions

whilst each roofshape is assigned a psuedo-random colour.

Figure 3.35: *close-up results of the maximal area roofshape segmentation applied in order to identify roofshapes in the city of Bath dataset*

Figures 3.34 and 3.35 provide qualitative indications of the performance of MARS. The close-up figure 3.35 (in particular) illustrates the separation of distinct roof-shapes for arbitrary building forms. However note that it also demonstrates that the segmentation is not perfect since there are still instances of over and under segmentation present. Nonetheless the majority of significant roof-shapes seem to be correctly isolated.

Distribution of Building Attributes: The following graphs record the distributions of building attributes in the city of Bath - calculated as a product of the segmentation method. The first depicts the spread of building sizes in the 1m point-spacing dataset, whilst the subsequent two depict the spread of roof-shape counts (cardinality) for buildings in the dataset and the distribution of roof-shape surface-area resulting from the area-maximisation step employed during MARS.

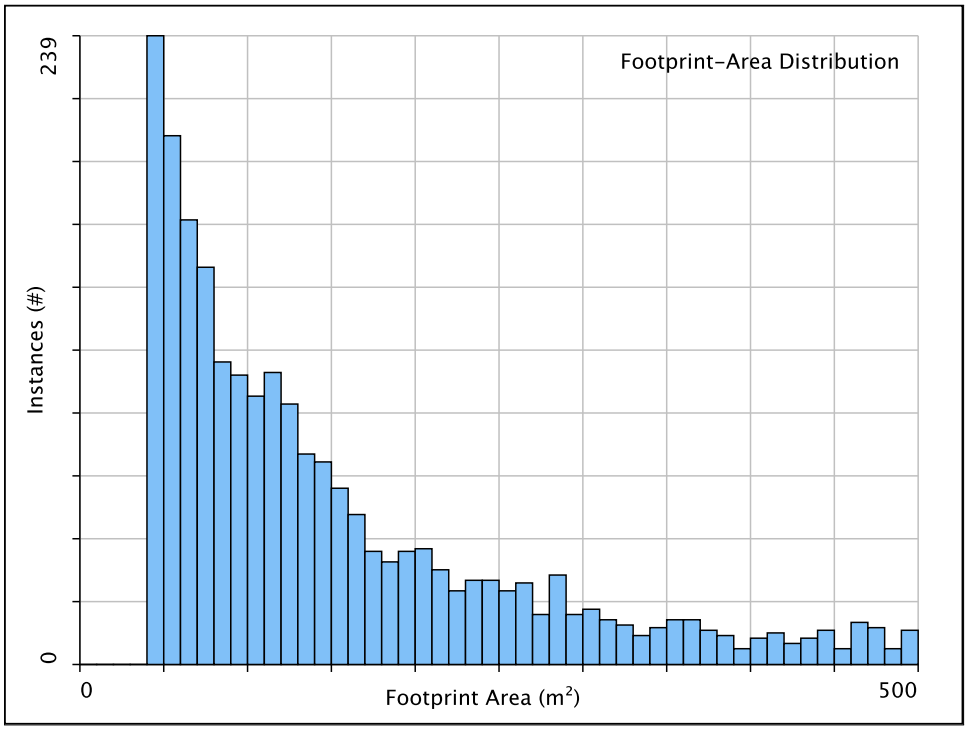

Figure 3.36: *bar-graph of the distribution of building footprint surface-areas in the city of Bath dataset - illustrating the non-linear relationship between the frequency of buildings and the footprint area*

Figure 3.36 indicates a non-linear (inverse exponent) relationship between frequency and building size. The interesting thing about this auxiliary analytic measure is that it provides a high-level city *signature* that could feasibly be used to compare and contrast distinct geographic regions based on the distribution of types of architecture present in each.

Figure 3.37 on the other hand considers the spread of roofshape attributes over the Bath dataset, and suggests that there is a propensity for buildings with four dominant roofshapes (right). Additionally note that although the user supplied minimum roofshape area is set to $10m^2$ there are still a handful of roofshapes that fail to meet this tolerance (left). Positively though it indicates that the proportion of roofshapes that could not be refactored by non-maximal suppression is negligable at 1m point-spacing, depsite the fact that there are still some non-conformals present.

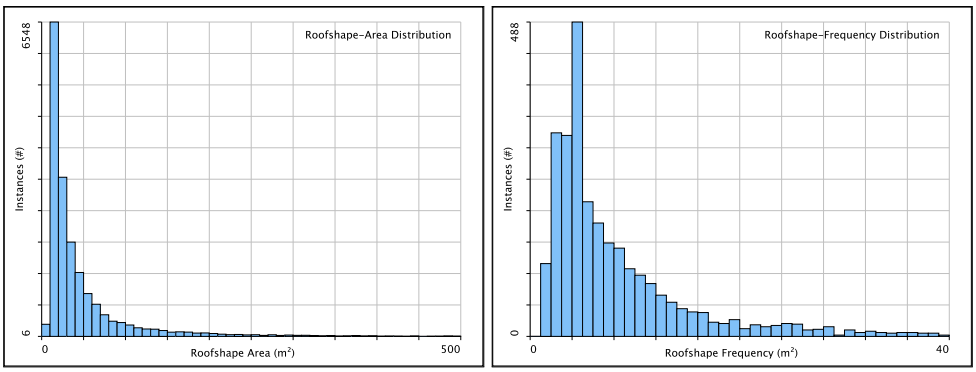

Figure 3.37: *bar-graphs indicating the distributions of (left) roofshape surface-areas and (right) the number of roofshapes per building, for the city of Bath dataset - illustrating (left) the proportion of non conformal segments (with the user supplied minimum roofshape area set to 10m^2) and (right) the most common roofshape frequencies as four and two.*

Growth in Execution Time: The following graph documents the growth in execution time of the segmentation stage over subsets of the city of Bath dataset. Each point in the line graphs represents the mean of 10 executions. This graph also documents the effect that the noise cancellation has on the segmentation runtime.

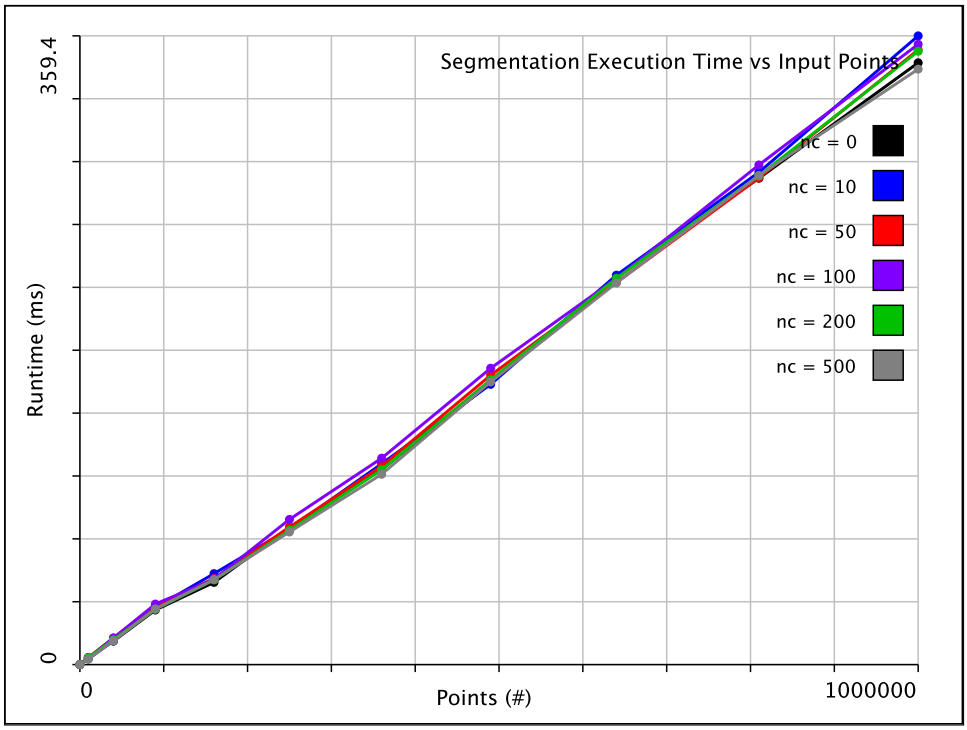

Figure 3.38: *growth in runtime for segmenting different sized subsets of the city of Bath dataset with varying noise-cancelling iterations applied*

Figure 3.38 constitutes a key result as it documents the almost perfectly linear growth in segmentation runtime relative to the number of points in the airborne scan for MAMMAL at 1m point spacing. Note: that the cost of noise-cancellation at this resolution is negligible even when the maximum number of iterations is high.

Vectorisation Results

The city of Bath vectorisation results document the performance of MAMMAL's 2D building footprint and roofshape detection. The key quantitative attributes profiled are the growth in execution time as a product of the size of the input pointset, the geometric accuracy of the resulting vector-shapes - in terms of the union over intersection and Hausdorff measures - as a product of varying the maximum error tolerances and the level of compression of the sparse vector shapes relative to their corresponding scan-converted dense vector shapes. These measures enable one to determine the extent to which MAMMAL is effective at producing accurate 2D vector shapes for automatic map updating.

Geometric Error : Hausdorf Distances, Intersect over Union Ratios: The response of the vectorisation stage in terms of the geometric error present in the output on-plan shape-nets is documented in figure 3.40.

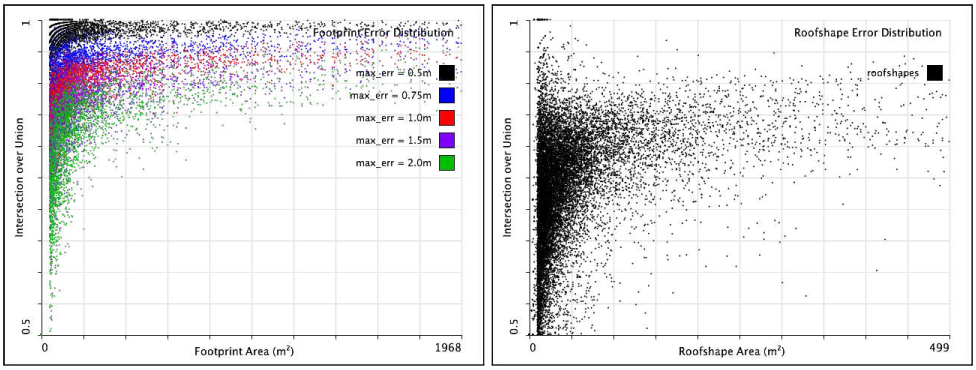

Figure 3.40: *scatter graphs of the error response for vectorisation of the city of Bath - documenting (left) per-building-footprint, and (right) per-roofshape error measures*

The scatter graphs in figure 3.40 document per-building (left) and per-roofshape (right) approximation error for the 1m Bath dataset - against each point-cluster's dense scan converted boundaries - relative to the surface-area of the shape using the Intersection-Over-Union Error Measure.

Level of Compression : Dense to Sparse vs Surface Area: The level of compression is documented in figure 3.41. The graphs plot the compression ratios for individual buildings (left) and roofshapes (right) for the city of Bath dataset.

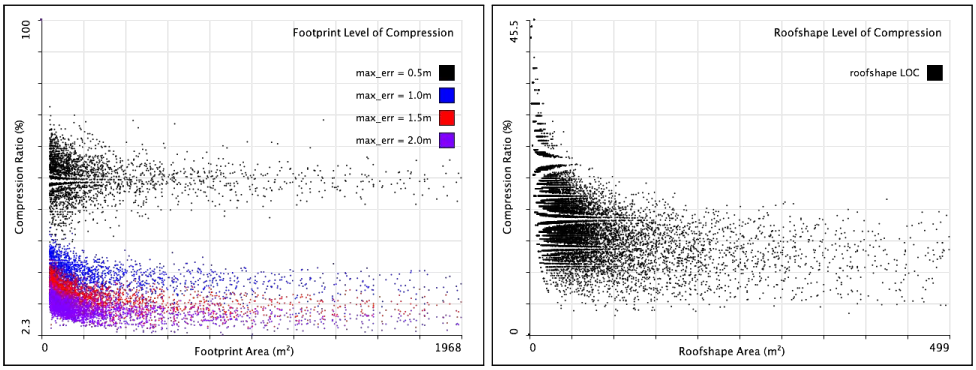

Figure 3.41: *graphs of the level of compression of the sparse building footprints and roofshapes constructed by MAMMAL's vectorisation for the 1m city of Bath dataset*

Growth in Execution Time: The rate at which the execution time of the vectorisation stage grows as a product of the number of input points is profiled in figure 3.42. It indicates the effect that the user supplied maximum error tolerance has on the time taken to vectorise the city of Bath dataset.

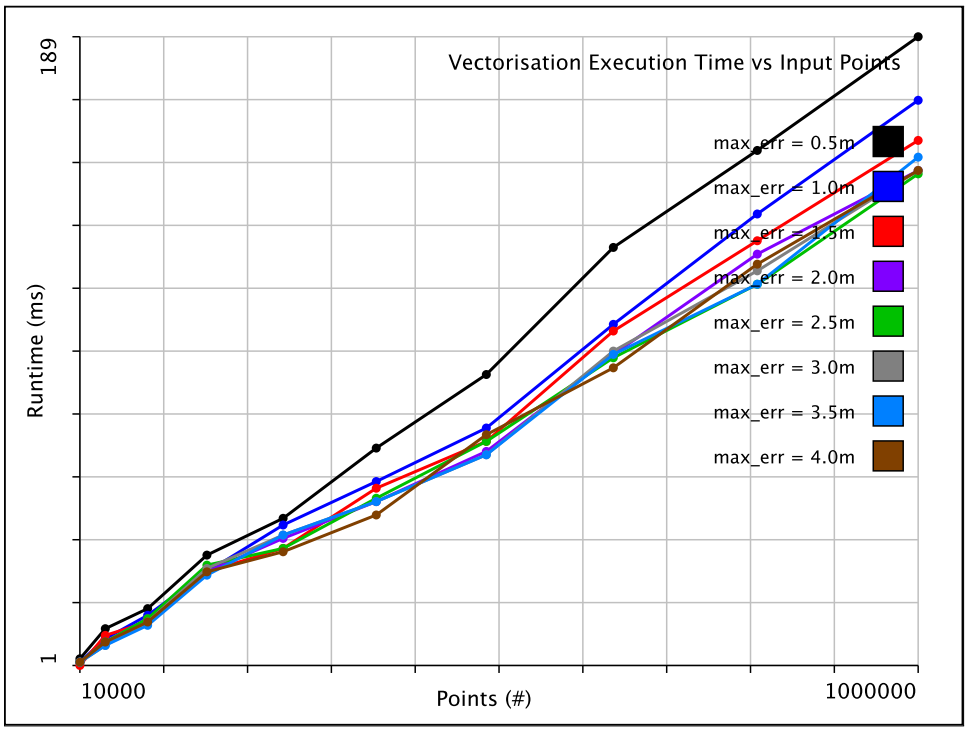

Figure 3.42: *growth in execution time for vectorisation of the city of Bath: indicating a linear relationship between the number of input points and the total execution time of the process - also denotes the relative increase in runtime incurred by stricter error tolerances.*

This key results demonstrates the linear growth in runtime of MAMMAL's vectorisation stage at 1m point-spacing. It also indicates the relative cost of demanding a stricter error tolerance at lower-resolutions. Note: that even at an impractically strict error-tolerance ($\pm$ 0.5m) the contribution to the growth in complexity is still linear - which indicates that the cost of the data-driven parametric approximation in 2D (as a product of error-tolerance) is a fraction of the overall vectorisation runtime.

Projection Results

These results document the performance of the roofshape projection stage - which maps the 2D shape-nets to 3D surface models. Again the important attributes are the geometric error of the output models the growth in execution time and the level of compression between the input range points and the output models.

Geometric Error : Root-Mean-Squared (RMS) Errors: The RMS error for projected buildings in the city of Bath dataset are documented in figure 3.43. The RMS error represents the average point-to-plane vertical distance between each sample point in the input point-cluster and a discretisation of the set of mass-models generated by the projection.

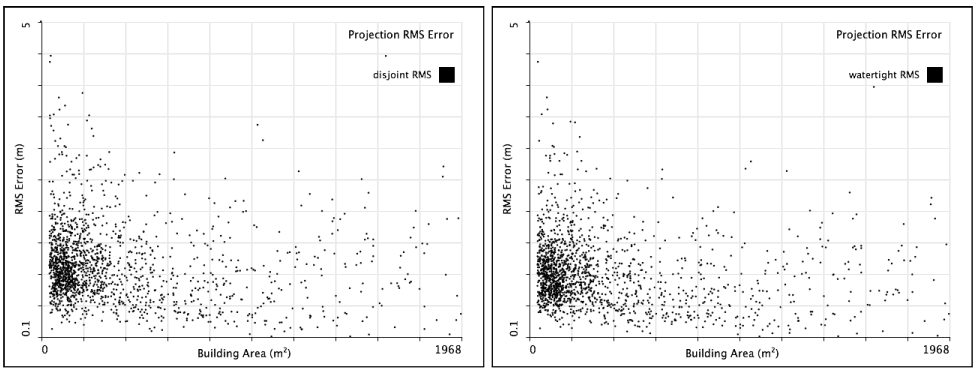

Figure 3.43: *scatter-graphs of the RMS error for projection of the city of Bath using (left) disjoint and (right) watertight roofshape-nets.*

The result indicates that there is a loose relationship between the resulting RMS error and the size of the building, which suggests that more points tend to make for more stable projections. However the nature of the result makes it difficult to draw a precise relationship between building size and expected RMS error. Additionally one should bear in mind that with the 1m point-data the proportion of noisy, boundary or anomalous points is greatest (relative to the 50cm and 25cm datasets documented subsequently) - which means that the scope for compromises to how robust the projection is at 1m is also greatest. Essentially there are fewer sample points per building at 1m point-spacing and as such one would expect any detrimental effect to be most significant. However (as stated) this is actually quite useful because it helps determine performance in the worst case scenario. Further there is a slight reduction in RMS error for the watertight (right) projected models, relative to the disjoint (left) projected models. This could be attributed to the fact that the disjoint models have the potential to introduce gaps between neighbouring roofshapes which add to the overall error because such points end up without corresponding surfaces which means there variance will be equivalent to their elevation. Due to the fact that such points occur primarily at the boundaries of roofshapes the overall detriment is limited and the reduction in error subtle. This indicates that any points not represented by the projected models will contribute the greatest amount to overall geometric error of the output.

Level of Compression : Input Point Count vs Output Vertex Count: The level of compression of the output projected models relative to the number of input points is reported in figure 3.46. The values are simply scalar ratios that represent the factor of input scan points to output model vertex.

Note in particular in figure 3.46 that at 1m point-spacing the output projected models may actually contain more vertices that the input points.

This indicates that for lower resolution scans MAMMAL cannot necessarily guarantee a sparser model than raw interpolation of the points. This can largely be attributed to the fact that in such instances any form of simplification has a greater impact on the accuracy of the result than at higher resolutions - and the fact that when there is little data - the representation of walls (vertical features) in the projections (which are not explicitly represented by the point-clusters) bears a higher relative cost - and can lead to a duplicative effect on vertex-count.

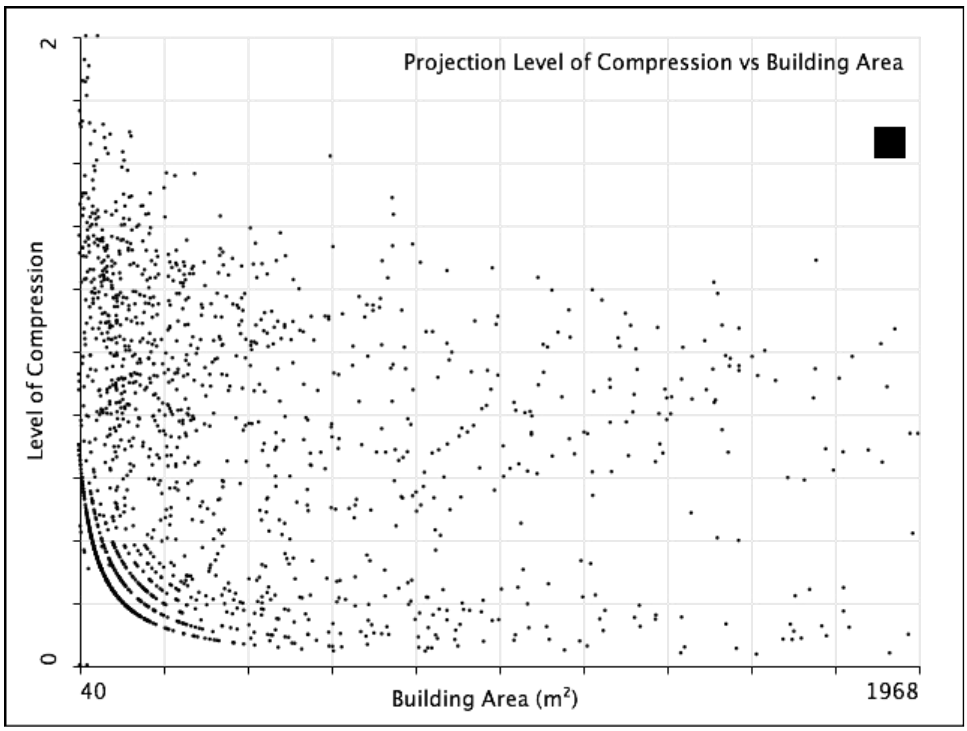

Figure 3.46: *the compression ratios of input points to output vertex plotted as a product of the extents (size) of buildings in the Bath dataset.*

Level of Compression : Graph of LOC vs RMS/Volumetric Error: These graphs document the manner in which the level of compression of each model influences/is-influenced-by the building's point-elevation error and volumetric error.

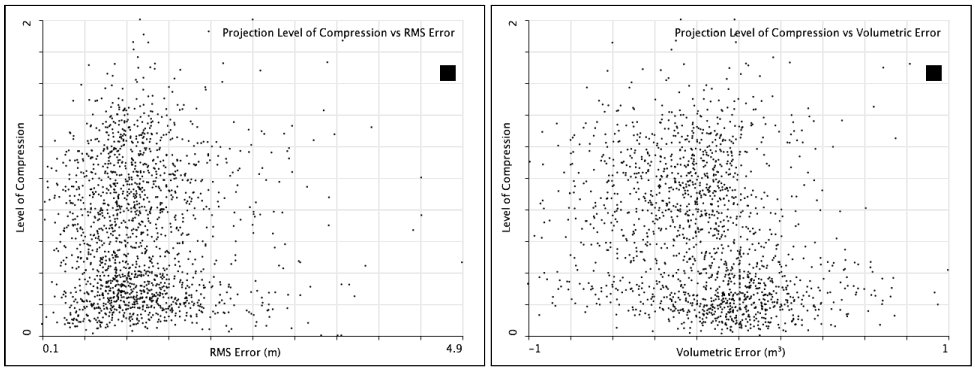

Figure 3.47: *scatter graphs of compression-level versus error for (left) RMS measures and (right) volumetric measures - for the Bath dataset.*

The spread of the measures in figure 3.47 is somewhat arbitrary and as such does not provide enough information to draw a precise relationship between a building's discrete (point to plane) and continuous (integral/volumetric) error at 1m point spacing and its typical level of compression. This suggests that for low-resolution scans the level of compression and geometric accuracy of masses produced by MAMMAL are not directly related to the sparseness of the representations generated.

Growth in Execution Time Graph: The rate at which the execution time of the projection stage increases as a product of the number of points in an input scan at 1m point-spacing is profiled in the graph in figure 3.48.

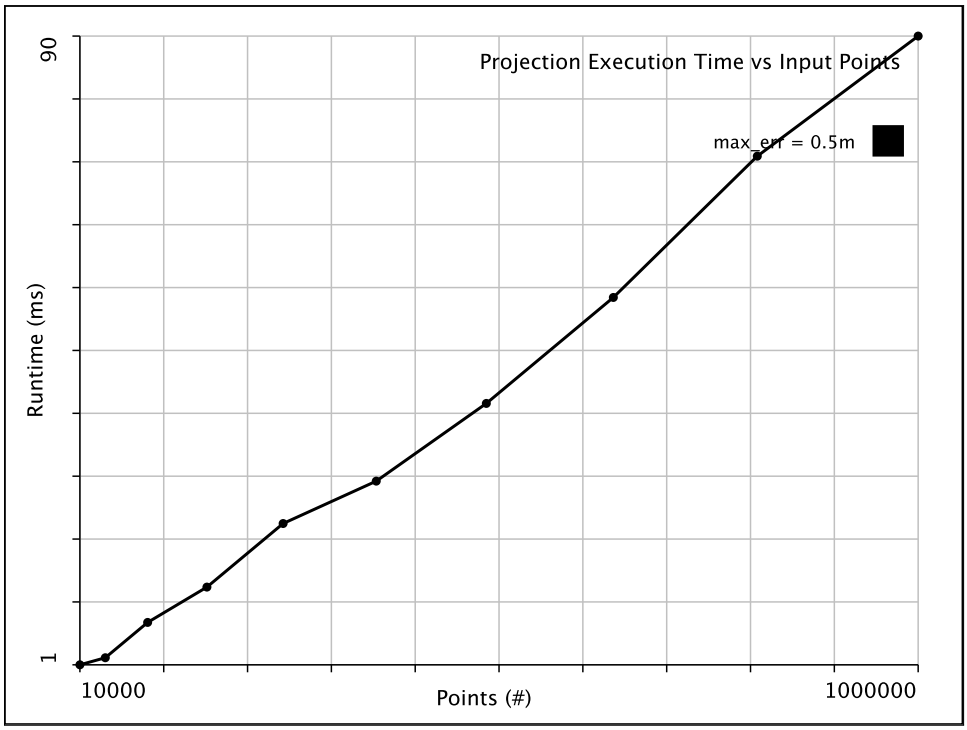

Figure 3.48: *plot of the growth in execution time as a product of the number of input points for the projection stage given 1m spacing point-data.*

This key result indicates that at 1m point-spacing the projection stage executes in roughly linear time, relative to the number of input points or using big-Oh notation: O(n) - where n is the number of input points. The reason this is important is because the computational complexity of an algorithm (in particular its growth in runtime relative to the size of the input) largely controls an algorithm's scalability.

Positively the cumulative results for the city of Bath indicate that for lower resolution scans MAMMAL's exhibits linear runtime growth - which is an important result in terms of its scalability. However they also indicate that at 1m resolution MAMMAL is more suited to producing LOD0 or LOD1 models and cannot produce physically accurate and highly detailed LOD2 models from such lower resolution data.

3.4.3 City of London - 4ppm (50cm)

Experimental Setup

The city of London experiments exploit digital surface and terrain elevation models of a region of the UK capital, covering roughly 10km x 10km. This set of experiments in particular represents vital test-cases since London is actually the city for which the idea of Automatic Temporal Updates was devised. Vitally though the London dataset is subject to greater sensing noise and partial and missing data than the city of Bath, and as such (despite the larger number of samples) provides a greater challenge for the MAMMAL algorithm in terms of the mitigation of undesirable sensing artefacts. The London dataset (as the Bath dataset) was acquired from the Geomatics group [47] and is in the form of a composite DEM.

Note: in order to keep the exposition of the results flowing - and (to an extent) to help minimise repetition - this subsection provides a summary of the pertinent aspects of the City of London results. This synopsis is largely to enable us to devote greater attention to the examination of the higher-resolution City of Manchester experimental results which follow.

Summary of City of London Results

The key observations of the City of London experiments are noted below.

- The performance of the segmentation improves as a product of the 50cm data. In particular the effects of non-conformal suppression are more distinctive - however the maximisation step also takes proportionally longer.
- The vectorisation performance improves massively with the 50cm data - however there remain subtle artefacts in the co-related shape refinement nets that still require addressing. Observation reveals that these result largely from irregular building complexes.
- The projection stage yields more coherent geometries given 50cm data than at 1m - but it still yields some models that look like automated geometry.
- The proportion of roof-shape components modelled by the non-linear optimisation increases with the higher resolution London data relative to the Bath data. Essentially the *take-up* of the parametrics over roof-shapes improves - however this comes at the cost of increased execution time.

3.4.4 City of Manchester - 16ppm (25cm)

Experimental Setup

The Manchester experiments exploit digital surface and terrain elevation models covering roughly 2km x 2km. The dataset is well sampled and exhibits fewer anomalies than the London dataset. However even at 25cm point-spacing there remain numerous unavoidable artefacts that have the potential to degrade the geometric performance of MAMMAL. These experiments delve deeper into MAMMAL's performance than with the preceding Bath and London datasets. In particular a number of additional tests are introduced that profile the resulting reconstructed models relative to open-access building shape data and alternative reconstruction methods. The key benefit of the higher resolution DEMs is that they enable consideration of MAMMAL's computational complexity as a product of the growth in execution time incurred as the density of input points increases. Additionally in practice a firm that produces Architectural and Engineering Visualisations will tend to opt for the highest resolution scan data that is available in order to minimise the introduction of approximation error. Fundamentally the Manchester dataset helps determine how well MAMMAL scales with point sampling density.

Segmentation Results

Growth in Execution Time: Figure 3.49 documents the rate at which the execution time of MAMMAL's segmentation grows relative to the number of points in the input scan at 25cm point-spacing. In particular note the increased runtime of the noise-cancellation stage for this higher resolution data relative to the Bath dataset.

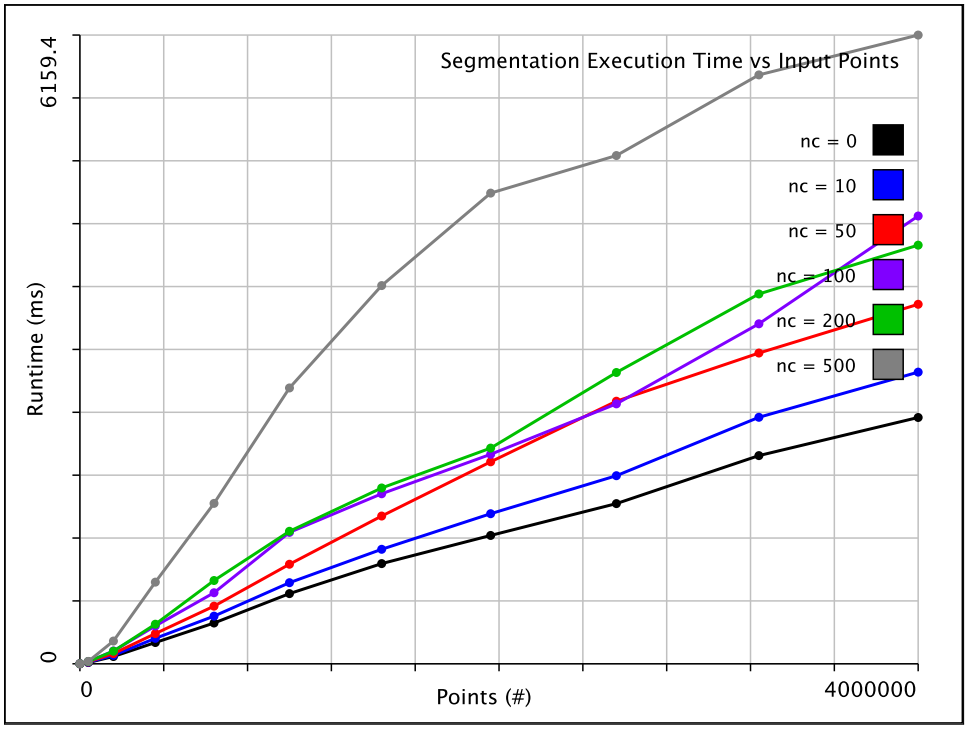

Figure 3.49: *growth in runtime for segmentation of buildings at 25cm point-spacing as a product of varying the maximum number of noise cancellation iterations employed by MARS*

The key observation in figure 3.49 is that at higher-resolutions noise-cancellation (non-maximal-suppression) contributes to a larger proportion of the overall segment runtime than at lower resolutions. In particular - the number of noise cancellation iterations required to resolve the roof-shape arrangement (so as to conform to the MAMP objective) is significantly more. However (despite this) the runtime still grows in a quasi-linear (or sub-linear) fashion (e.g. at max 500 iterations).

Figure 3.50 documents the result of processing an irregular building to provide a qualitative sense of MARS' segmentation performance at 25cm point-spacing.

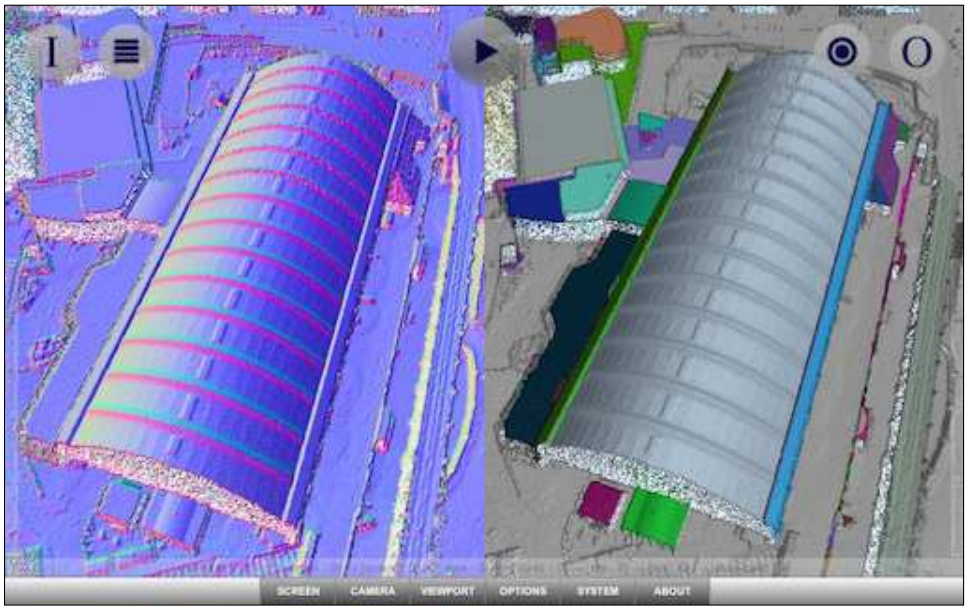

Figure 3.50: *result of segmentation of a particularly challenging building instance (exhibiting curvature) at 25cm point-spacing - illustrating (left) the input points coloured by normal and (right) the isolated clusters coloured by segment group*

The positive aspect of this is it demonstrates MAMMAL's ability to decompose regular (planar) and irregular (non-planar) components. In particular observe the manner in which the semi-cylindrical component is handled. Even with the ribbed edges along its surface, MARS is able to correctly isolate this component.

Vectorisation Results

The city of Manchester vectorisation results document the performance of MAMMAL's 2D building footprint and roofshape detection. These results follow the same experimental conventions as the city of Bath and city of London datasets.

Geometric Error : Hausdorf Distances, Intersect over Union Ratios: The response of the vectorisation stage at 25cm point-spacing (in terms of the geometric error present in the output on-plan shape-nets is documented in figure 3.51.

The result demonstrates a clear relationship between the continuous error and the surface-area of vectorised components - with smaller components contributing greater error. In essence one can state that the larger the roofshapes (typically)

the greater the integral fit relative to the dense scan converted boundaries.

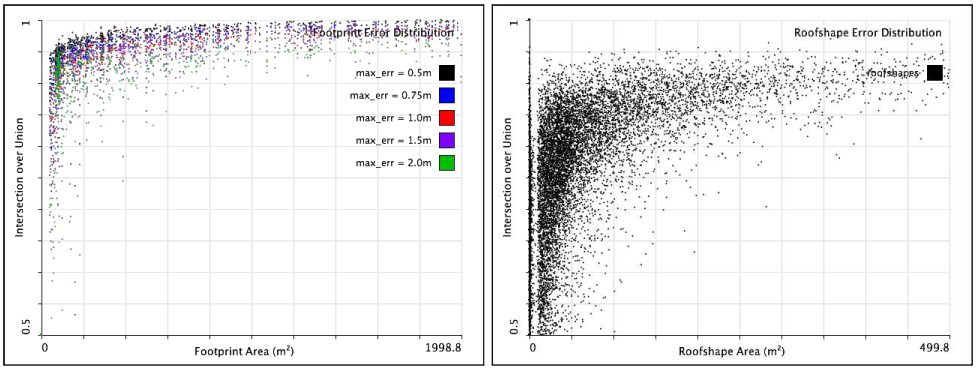

Figure 3.51: *graph of error response for vectorisation of the city of Manchester - documenting (left) per-building-footprint, and (right) per-roofshape error measures*

This result confirms one's intuition in that the larger each roof-shape the less significant the contribution of error around its boundary. However note also that roof-shape error typically exceeds footprint error.

Level of Compression : Dense to Sparse vs Building Area: The relationship between the sparseness and accuracy of vector-shapes generated by MAMMAL at 25cm point-spacing is documented in the plots in figure 3.52.

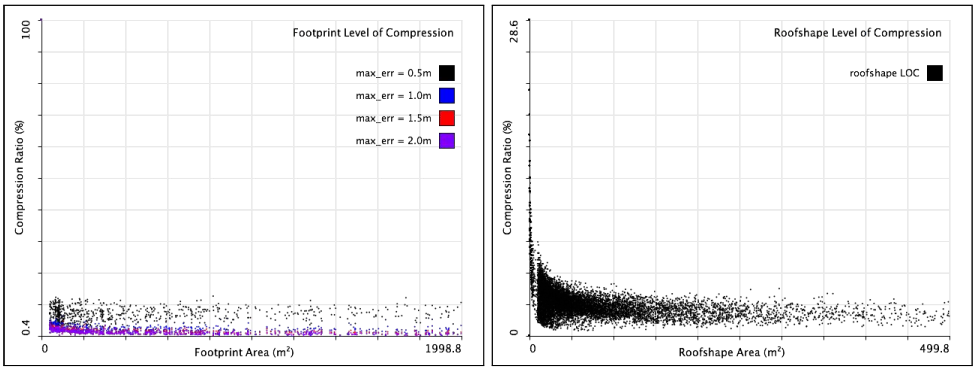

Figure 3.52: *graph of compression response (sparseness) for vectorisation of 25cm airborne scans - documenting (left) the effect of varying the max-footprint-error tolerance and (right) the per-roofshape measures*

Observe in figure 3.52 that the behaviour of the vectorisation stage (in terms of its sparseness whilst conforming to an error tolerance) improves vastly for the 25cm data relative to the 1m and 50cm data. This is reasonably obvious as it confirms the intuition that the higher the resolution of the input the greater the scope for compression. Essentially more points equate to greater scope for simplification.

Projection Results

The Manchester projection results document the performance of MAMMAL's data-driven 2D to 3D *parallax pop-up* stage for 25cm resolution scans. These results follow the same experimental conventions as the Bath and London datasets.

Geometric Error : Root-Mean-Squared (RMS) Error: The RMS error for the pro-

jected buildings in the Manchester dataset are documented in figure 3.53.

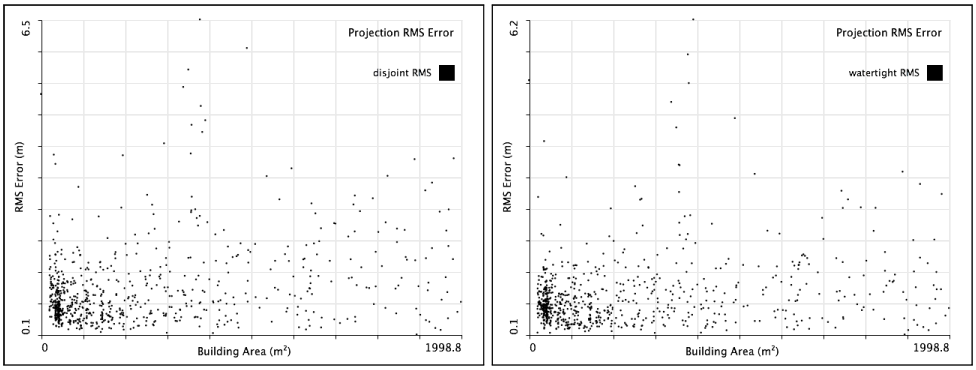

Figure 3.53: *RMS errors for buildings in the 25cm city of manchester dataset when (left) disjoint, and (right) watertight roof-shape-nets are used*

Note in figure 3.53 that whilst most buildings yield a tight discrete fit - the presence of erroneous clusters (i.e. vegetation points that are mis-classified) offsets the distribution of errors in the scene. This indicates that a handful of the objects can possess mis-represented regions that (as such) yield greater error.

Beyond mis-represented objects - the increased resolution of the points means a greater number of boundary positions can influence the overall error of a building.

Geometric Error : Volumetric Mass Error: The volumetric error for projected buildings in the Manchester dataset are documented in figure 3.54.

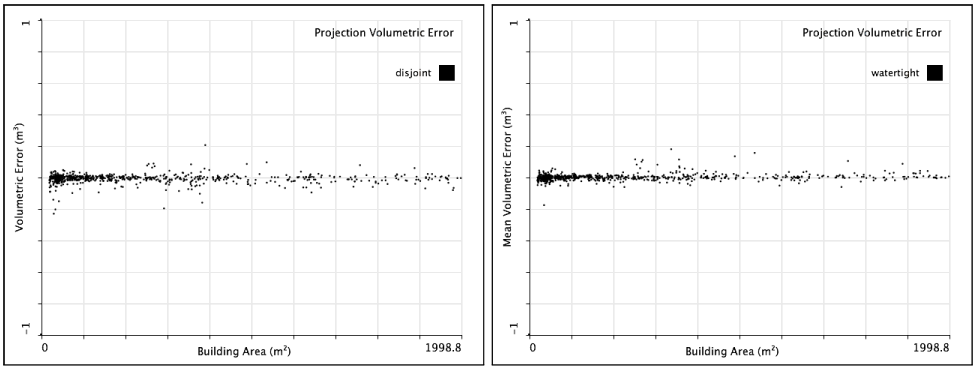

Figure 3.54: *the volumetric error for the projections in the 25cm Manchester dataset - indicating (left) disjoint and (right) watertight measures for projected building masses*

The key observation in figure 3.54 is the decrease in volumetric error for the higher resolution 25cm data relative to the lower 1m and 50cm resolution datasets. This suggests the volumetric fit of each building improves significantly as the density of the input increases. There is far less variation in the result. Interestingly this also indicates that though the point-to-plane error for misrepresented objects can exceed the error-tolerance - generally there is less of an effect on the volume of a representation which more accurately characterise the extents of each mass.

Level of Compression : Input Point Count vs Output Vertex Count: The relationship between sparseness and building extent for the projected masses in the 25cm resolution Manchester dataset is documented in figure 3.55.

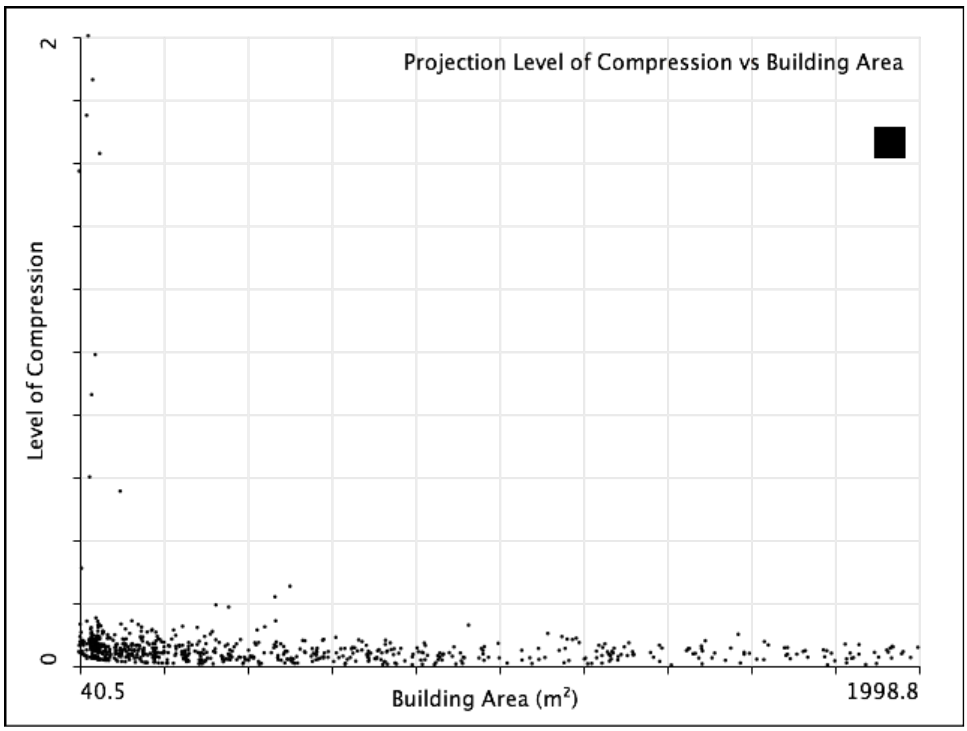

Figure 3.55: *level of compression for data-driven projected masses at 25cm resolution*

Figure 3.55 indicates a more uniform relationship between building size and sparseness at 25cm than at 1m and 50cm. However observe that for very small buildings (i.e those near the minimum area tolerance) - such as small residential masses - MAMMAL may still use more vertices to represent a building than are present in the input point-cluster, even for 25cm data. Fortunately though figure 3.55 indicates that such instances are relatively rare - and that generally the compatification of the resulting masses is reasonable (typically less than 10% of the input).

Level of Compression : Graph of LOC vs RMS/Volumetric Error: The relationship between the accuracy and brevity of MAMMAL's projected masses for 25cm resolution airborne scans is documented in figure 3.56.

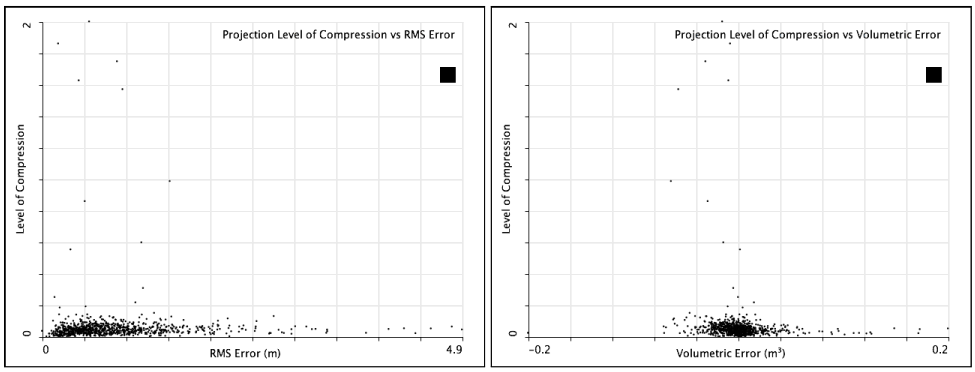

Figure 3.56: *scatter graphs of compression-level versus error for (left) RMS measures and (right) volumetric measures - for the Manchester dataset.*

Figure 3.56 indicates a more stable correspondence between error and compat-

ification for the projected masses produced by MAMMAL given 25cm resolution scan data. The spread of errors is less diverse than at lower resolutions. However the relationship is looser than that between compatification and building size.

To complement the projection results for the Manchester dataset figures 3.57 and 3.69 provide a qualitative view of the outcome of the process.

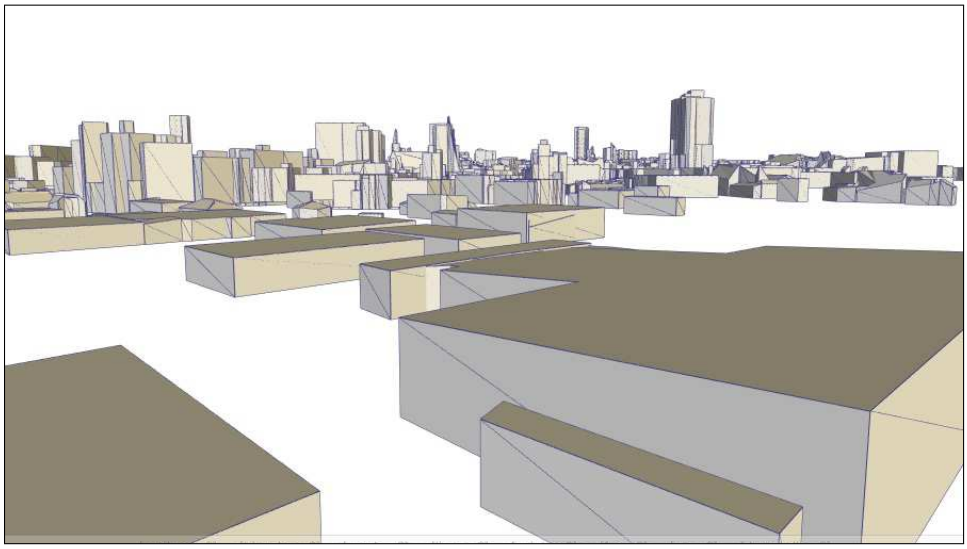

Figure 3.57: *city-scale massing models automatically generated by MAMMAL's linear roof projection routines for the 25cm Manchester dataset - with the camera positioned on a building's roof to simulate a first-person view-point of the city - from a specific location*

Figures 3.57 and 3.69 document the appearance of the linear and non-linear data-driven projections constructed by MAMMAL as a signpost to the level of detail that is attainable. The main observation is that these assets are relatively close to manually created models in terms of how clean and compact they are.

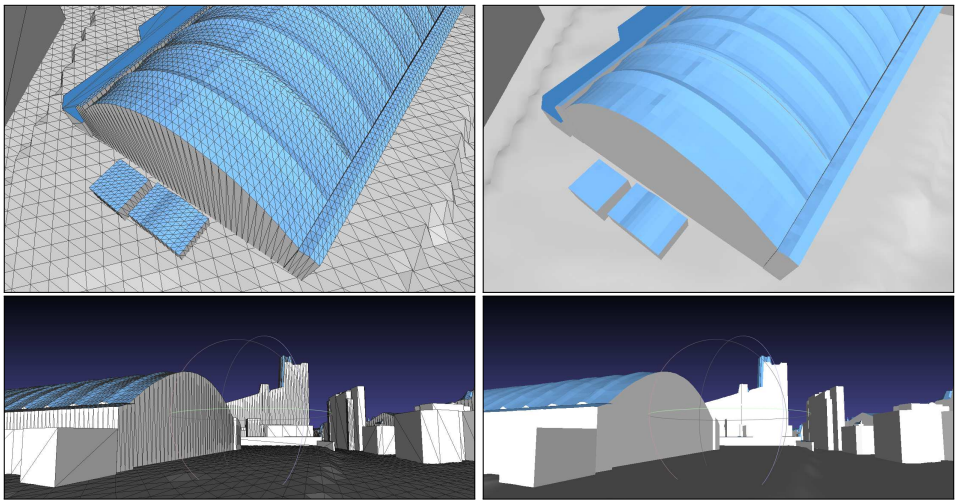

Figure 3.69: *non-linear roof projection using MAMMAL's structured sub-divisions - illustrating (top-row) a close-up of the roof-mesh for the building in figure 3.50 - and (bottom-row) the building at ground-view level - note: the terrain is a sub-sampled DTM interpolation*

Optimisation Results

The optimisation results document the computational performance of the non-linear parameterisation of building point-clusters in the Manchester dataset.

Growth in Execution Time: Figure 3.70 documents the growth in runtime for MAMMAL's optimisation stage on the 25cm Manchester scan data relative to the growth of the segmentation and vectorisation stages.

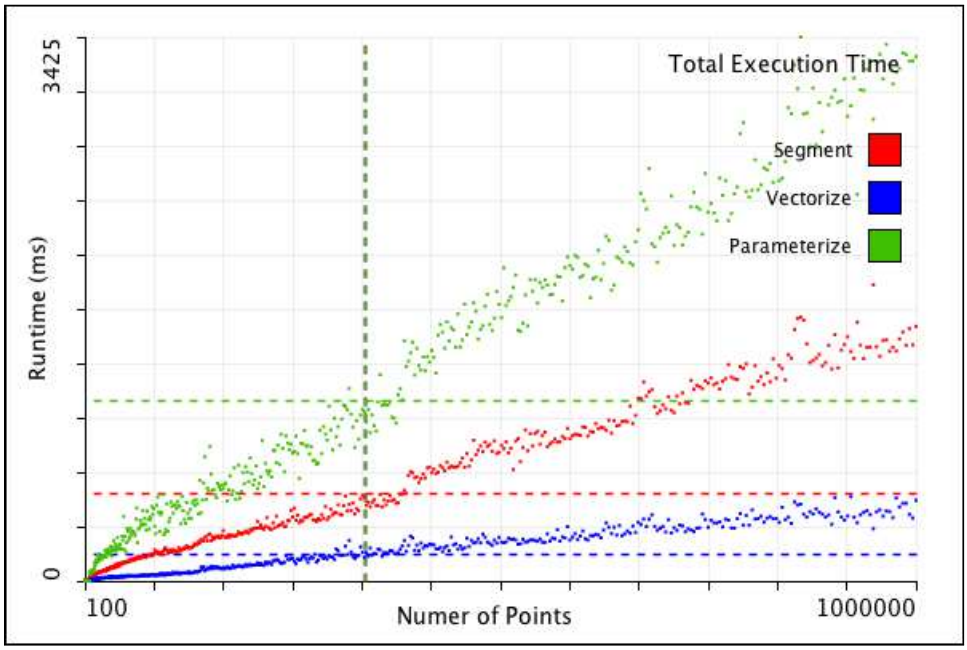

Figure 3.70: *growth in runtime for MAMMAL's procedural optimisation stage (at 25cm point-spacing) - relative to its segmentation and vectorisation stages*

Figure 3.70 documents the growth in execution time for the key stages in MAMMAL relative to one another. The main observations are that parameterisation is the most expensive sub-process - however (vitally) it still exhibits roughly linear runtime growth as a product of the number of points in the input.

Figure 3.71 illustrates close-up views of some of the optimised building components in the Manchester dataset to provide some qualitative context.

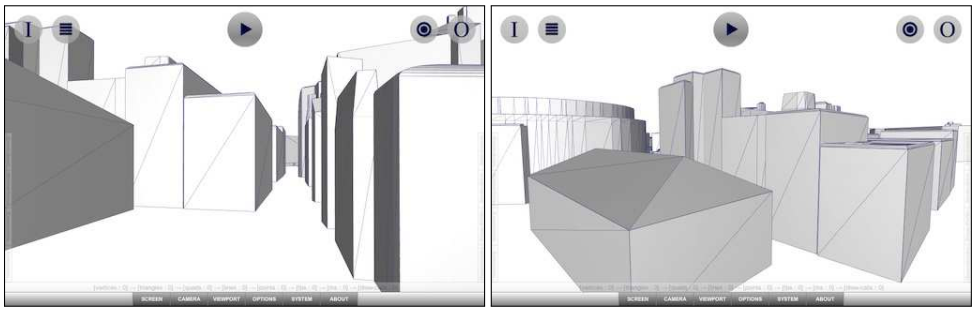

Figure 3.71: *close-ups screenshots illustrating the results of MAMMAL's automatic non-linear procedural optimisation of the 25cm Manchester dataset (zoom-in to-inspect)*

Post-Reconstruction Model Validation: Parametric Building Editor

MAMMAL's post-reconstruction interactive editor is an extension to the fully automated implementation that supports component level parametric alterations to the buildings in a reconstructed scene. The primary purpose is to enable a technician to refine the results of optimisation by changing the parameter values for individual building roofshapes in realtime. A technician selects part of a building (by clicking upon it) in order to display and manipulate a 2D parametric widget that instantly re-models the building component using the updated parameter values.

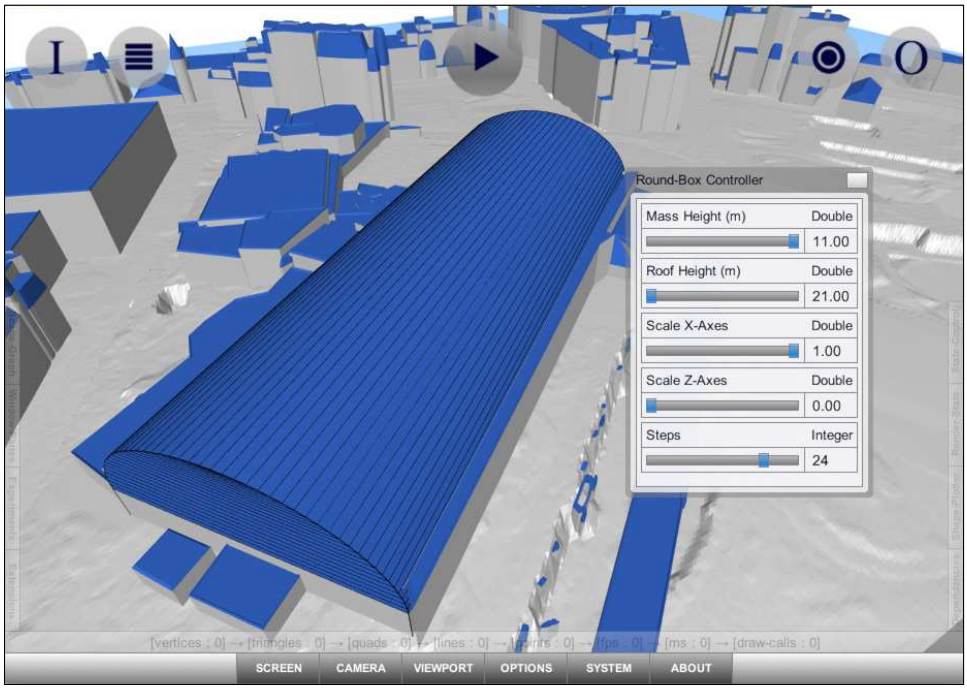

Figure 3.72: *post-reconstruction parametric model validator with optimised masses from the city of Manchester loaded and a 'round-box' instance selected for editing.*

Although this was written as an extension to the core MAMMAL implementation, what it demonstrates is a simple prototype of procedurally generated mass-models being parametrically (interactively) controlled in realtime automatically from reconstructed point-cloud data. Yes - you read correctly! For the surprising (and to some extent revolutionary) hidden benefit of MAMMAL is that it implicitly lends itself to the problem of recovering semantically-rich parametric descriptors automatically from airborne laser-scans. The parametric editor exploited as a post-reconstruction validator stands as evidence of this. Whilst this initial version leaves much to be desired - (especially in terms of the tiny set of functions exploited), it still represents far greater semantic richness than any reconstructive pipeline present today.

Essentially the truly novel aspect of the MAMMAL algorithm's procedural optimisation is that it automatically generates interactive parametric representations of city scale airborne laser scans. that allow high-level manipulation of the structure of components in each building with the use of semantic parametric handles.

The vital insight in this is the duality of the functional descriptors extracted by MAMMAL's non-linear procedural optimisation - i.e. the fact that the same un-

derlying representation is used to serve backward-chaining (reverse engineering) and forward-chaining (user-centric) model construction and manipulation.

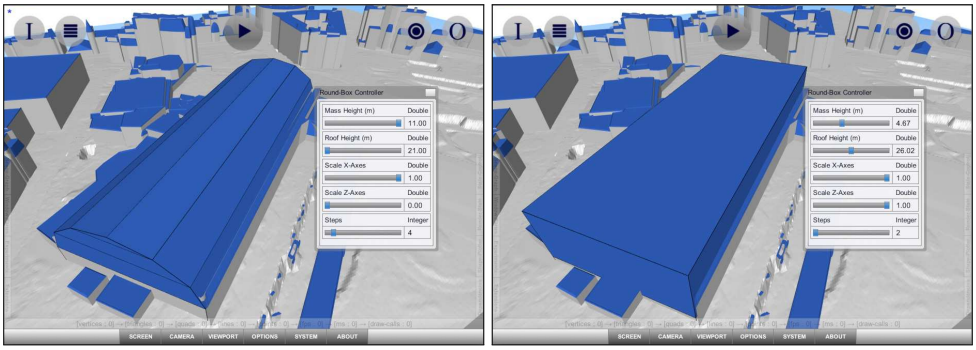

Figure 3.73: *post-reconstruction user-centric alterations made (via interactive parametric manipulation) to the optimised component depicted in figure 3.72 - coordinated in realtime*

For example figure 3.73 illustrates an end user altering the arguments to the automatically optimised *round-box* mass-model instance (depicted in figure 3.72) - in order to change the *grain* of the piecewise approximation of curvature encoded by the function. All of this occurs in realtime. This ability is simply a product of the fact that for each optimised component both an explicit and implicit representation exists. The explicit representation takes the form of a polygon-mesh, whilst the implicit representation is simply the name of a function and a list of the input arguments. Whilst most technicians will be interested in the explicit representation, those with procedural plans will lean towards the implicit representation. This preference is manifested in the parametric editor which simply instantiates and binds geometric model generator widgets using the parameter values of the optimised components in the reconstructed dataset. This is a key result in that it documents an advantage of MAMMAL that is currently unmatched by pre-existing methods.

In particular - even the highly regarded (dominant pre-existing) methods of Zhou and Neumann [165][166][167] and Lafarge, Mallet et al. [71][72][73][68] do not facilitate this level of semantic alteration to a reconstructed airborne scan dataset.

Note on Interactive Post-Processing: The vital property of the interactive validatory extensions to MAMMAL is that they are post-processing tools. They do not actually alter the reconstructive requirements. Instead they simply enable an end-user to select (switch between) alternative-mass-models instances or dynamically revise the parametric representations. The benefit of this is that they do not mandate any alterations to the automatic implementation. Rather they present the output in a manner that enables intuitive revision. That is all. In principle, if the quality of the automatic result is sufficiently high, then these extensions can be considered surplus to requirements. However in practice there are often trouble-case buildings that must be treated in isolation because their error-minimisers (best according to selection-criteria) are lacking aesthetically. These post-processing tools facilitate such alterations without having to re-process an entire site. Nonetheless the better the automatic result, the less the need for intervention.

Supplementary City of Manchester Results

Throughout this chapter you'll have undoubtedly noticed a propensity to use buildings from the city of Manchester as examples in the explanation of certain key concepts. This is largely down to the fact that it represents the highest resolution off-the-shelf data currently available. It provides the clearest indication of the structure of the underlying scene and is therefore closest to the sort of dataset that would be used in an industrial setting. As a product of this, during the course of development and experimentation, a number of additional tests were executed on the city of Manchester, the results of which are included in this auxiliary subsection.

Preliminary Comparisons with Pre-Existing Algorithms

To analyse MAMMAL's performance relative to pre-existing reconstruction algorithms, corresponding models were reconstructed using a variety of common general-surface and domain-specific reconstruction algorithms including: Marching Cubes, RANSAC, 2.5D Dual-Contouring, Poisson Reconstruction and direct height-map interpolation (as a baseline). The tables in figures 3.58, 3.59 and 3.60 document the results of the comparative evaluation. Figure 3.59 quantifies the attributes of each evaluated method for a single invocation of each - enabling one to compare measure for measure the performance of each. Figure 3.60 illustrates the qualitative differences between the pre-existing methods and MAMMAL's projection (ALV raw) and optimisation (ALV parametrics) methods.

Algorithm	1ppm	4ppm	16ppm
Interpolate (Baseline)	0.5048613	0.4762369	0.4476920
Marching-Cubes	4.5756348	4.0543478	3.6088912
2.5D Dual-Contouring	0.4321292	0.1309045	0.0304963
RANSAC	0.3248714	0.2454567	0.0397031
Poisson-Reconstruct	0.6841128	0.4565430	0.3457325
ALV-Raw	0.1804222	0.1086362	0.0108867
ALV-Parametrics	0.1460385	0.0759824	0.0304236

Figure 3.58: *table of the level-of-compression (LOC) values at different resolutions (point-spacings) for the algorithms evaluated in the comparisons with pre-existing techniques - each value specifies the ratio of the number of input points to the number of vertices in the output model :* $|out|/|in|$.

Note: interpolation refers to simply stitching together a gridded height-map and acts as a dense baseline. Both marching-cubes and poisson-reconstruct refer to meshlab's[19][20] implementations and signal the performance of general surface reconstruction methods. The method 2.5D dual-contouring refers to Zhou's hermite-grid based strategy[165] - and in particular the open-source implementation provided on his homepage [168]. For researchers looking to reproduce the behaviour the parameters used were: GridLength: 1.0, AcceptNumber: 4, RelativeDistance: 100.0, RelativeZ: 1.0, Weight: 1.0, ErrorTolerance: 1.0, SingularTolerance: 0.15, WallRectangle: 0, AntiNonManifold: 1, SnappingErrorTolerance: 1.2, SnappingMinimumLength: 7.0. Together 2.5D dual-contouring and ransac represent examples of commonly employed domain specific building reconstruction techniques.

	RMS Error (meters)	Runtime (seconds)	Vertices Used (#)	Triangles Used (#)	Mean Tri. Surface Area (meters^2)	Mean Volumetric Error (meters^3)	Size On Disk (megabytes)
Interpolation	0.157	1.2	1,790,768	3,476,204	0.139	0.0230385923	213.3
Marching Cubes	0.322	227.6	14,435,565	4,811,855	0.092	0.0457937666	843.7
Dual Contouring	0.493	97.1	121,985	243,095	2.964	0.0312454145	9.8
RANSAC	1.524	546.1	158,812	79,406	8.153	0.0966505392	8.3
Poisson Reconstruction	0.732	1140.8	1,382,930	2,773,383	0.549	0.1258947345	37.6
Our Method Raw	0.367	15.5	43,547	16,460	41.379	0.0269106868	1.6
Our Method Parametrics	0.908	28.7	121,692	42,288	29.786	0.0635228831	5.4
1st 2nd 3rd							

Figure 3.59: *quantitative results of reconstructing a 0.5 km^2 subset of the City of Manchester at 25cm resolution - comparing existing methods and MAMMAL. All models were derived from 2000x2000 Esri DEMs and stored in the Wavefront OBJ format.*

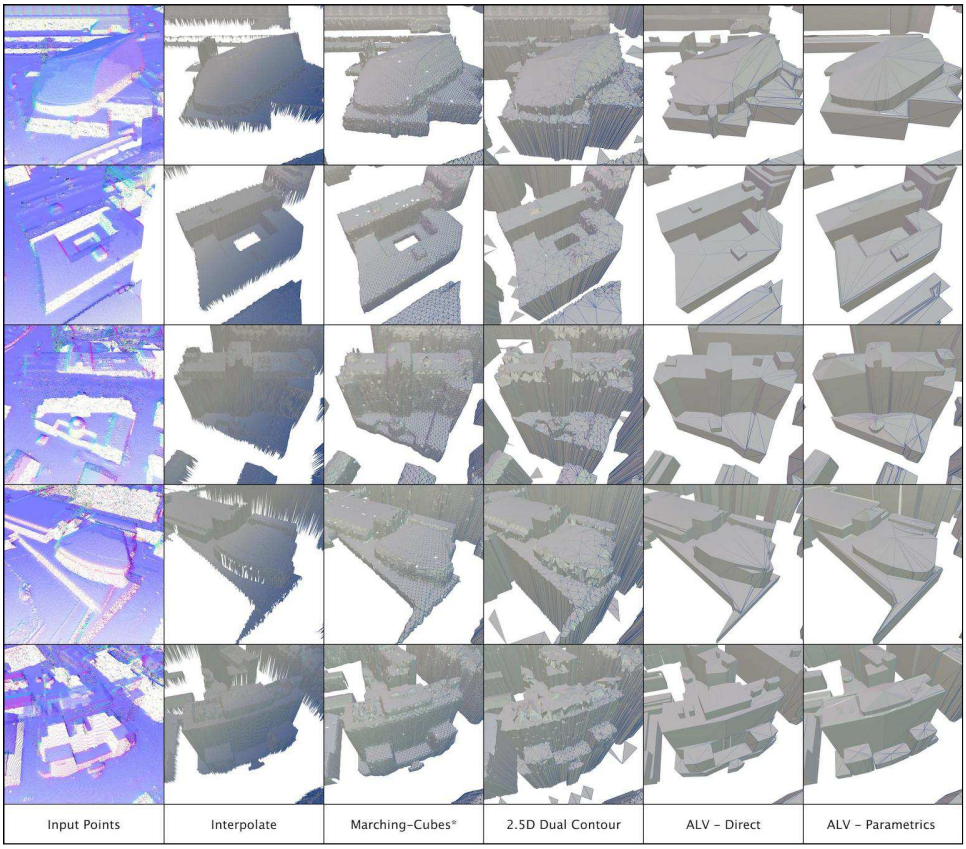

Figure 3.60: *qualitative results: comparison with pre-existing methods for figure 3.59. *→decimated-for-rendering*

Comparisons with Open-Street-Map Vectors

Evaluation relative to openly available datasets such as OSM building footprints provides further insight into the accuracy of the building segmentation and vectorisation. Figure 3.61 illustrates the deviance between Open-Street Map (OSM) building footprints and footprints recovered by the MAMMAL algorithm. The corresponding quantitative error measures are provided in figure 3.63. Further figure 3.62 illustrates close-ups of the deviances for a selection of irregular buildings - comparing the number of vertices used by MAMMAL (with a user-supplied maximum footprint error tolerance of 2m) to the number present in the OSM footprint.

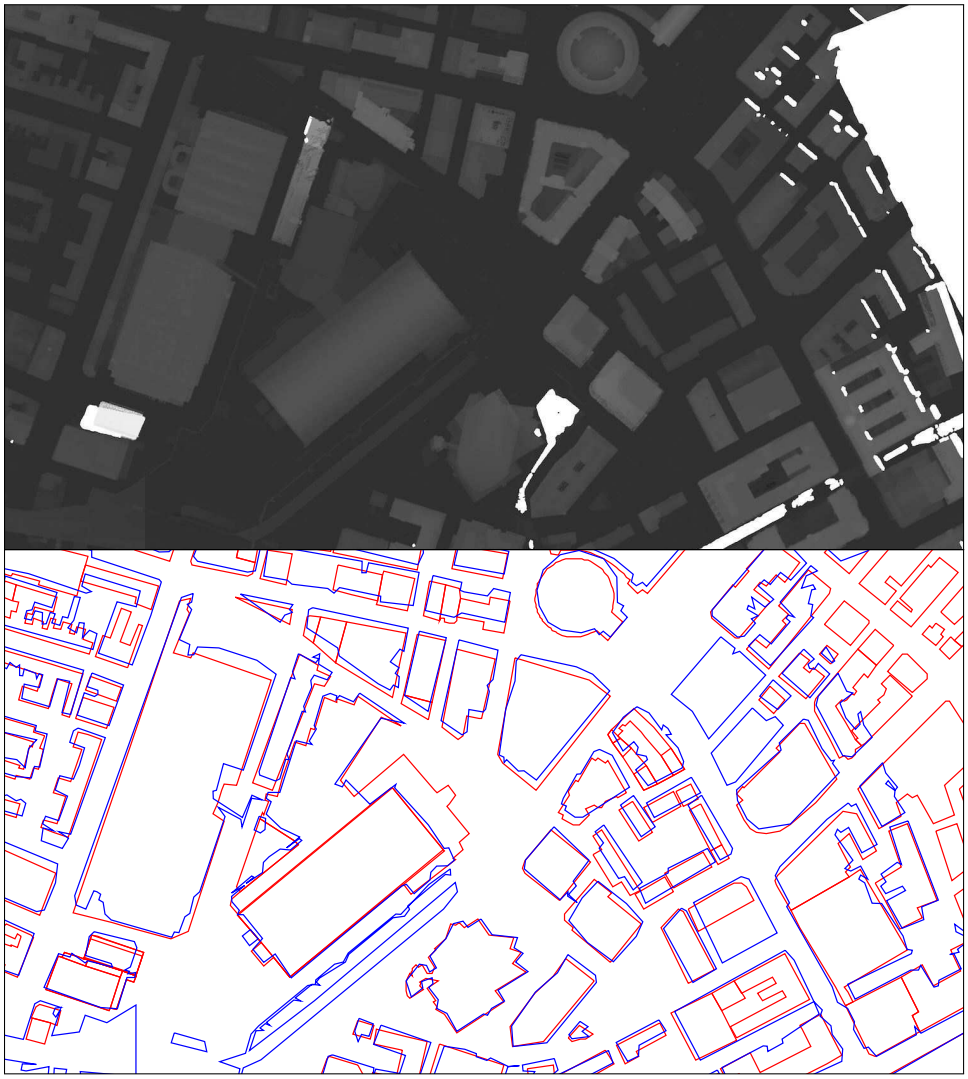

Figure 3.61: *comparison with open-street-map building-footprints (with vector data recovered by the MAMMAL algorithm in blue, and OSM in red), the input surface-model range-scan is shown above for reference*

In particular note in these figures the similar use of *maximal-length* edges in the characterisation of planar facade walls (edge-length maximisation).

Within these figures the OSM data is indicated by red vectors whilst the outcomes of airborne LiDAR vectorisation are denoted by blue vectors. ∩/∪ refers to the union over intersect error-measure whilst | → | symbolises the maximum-point-to-boundary-distance Hausdorf error-measure.

Critically there are two unavoidable sources of error in the evaluation with OSM. Firstly temporal variations between the datasets. Secondly the presence (or lack thereof) of features at varying scale spaces (left figure 3.62).

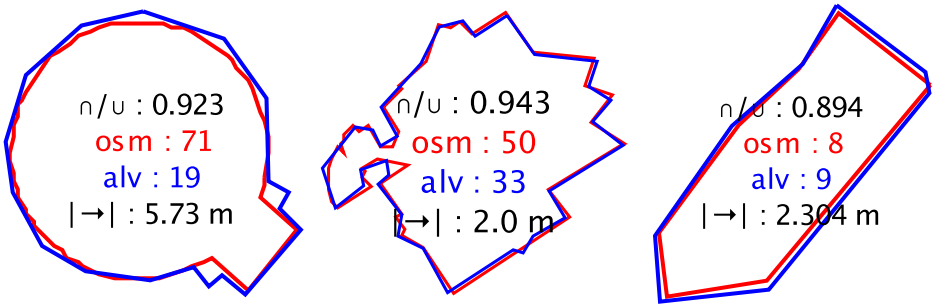

Figure 3.62: *comparison with OSM building-footprints for irregular instances - illustrating the number of vertices used to model each footprint and the associated error values between the automatic vectors and the OSM vectors*

Observe that the presence of two additional corners represented by the scan-data but omitted from the OSM (in figure 3.62 for the left-hand footprint of the Manchester public-library referenced throughout this chapter) significantly increases the Hausdorff error. This is a manifestation of the secondary cause of unavoidable error between the automatic vectors and the OSM vectors. However even in the presence of variance caused by featural differences the interior fit (measured as ∩/∪) still represents a stable correspondence between ALV and OSM. The crucial aspect is that the quality of the fit of each shape relative to previously unseen manually constructed vector shapes is surprisingly high. This demonstrates that MAMMAL can be used to produce 2D-mapping building boundaries with a high-degree of accuracy whilst preserving features at variable spatial scales.

Hausdorff Error (m)	Intersect/Union	Compression Level
2.88613750	0.868114671	0.79233930

Figure 3.63: *mean error values for building footprint vectors in figure 3.62 with an input max-footprint-error of 2m - relative to the open-street-map footprints, for the 25cm point-spacing Manchester dataset*

Figure 3.63 quantifies the accuracy of the automatic footprint vectors relative to the OSM vectors. Note that whilst the maximum simplification error for the reconstruction is set at 2m - this tolerance only controls the deviation between each dense scan-converted boundary and its corresponding approximate sparse vector. Nonetheless the typical boundary error between the unrelated OSM dataset and the automatic vectors is just under ±3m. When one considers the additional sources of error between the representations (in particular the presence or omission of features that can offset the mean of the boundary error measure), this is more easily understood. The positive aspect is the interior (∩/∪) measure which indicates that the mean per-building pairwise-overlap of the sparse vector shapes

produced by MAMMAL relative to vectors from an alternative medium is over 85%.

Realtime Interactive Visualisation

Finally, to wrap up the auxiliary city of Manchester results, the outcomes of visualisation are documented. Figures 3.64 and 3.65 illustrate aspects of the visualisation of the building models recovered by MAMMAL.

Recall one of the key design goals of MAMMAL is to produce building geometry that is suited to interactive rendering and simulation.

Figure 3.64: *the city of Manchester rendered interactively in the Unity game-engine - illustrating an advantageous property of the proposed reconstruction operator - automatic construction of accurate real-time-ready architectural mass-models*

At a high-level this demands two things. First that the building models are compact enough to be rendered at scale on modern hardware and secondly they are aesthetically viable as building representations. In essence they must be light-weight enough to render in realtime at interactive frame rates of anywhere between 24-120fps - and they must look like structured representations of the built environment.

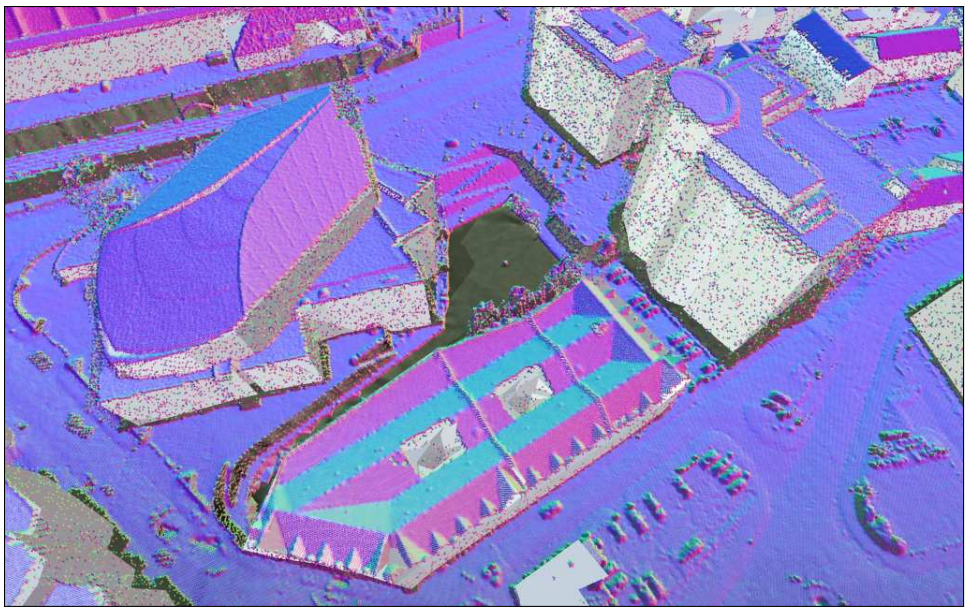

Figure 3.65: *close-up of models recovered using the sparsest (minimum-vertices) under error tolerance selection-criteria, with the corresponding input surface points rendered (for reference) with a normal-to-colour shader*

Figure 3.64 depicts the reconstructed Manchester model with a first-person perspective camera - as a signpost to the visual quality of the geometry produced by MAMMAL. Due to the sparseness of the resulting models interactive rendering is a breeze. Additionally each mass-model is consistently oriented (no flipped faces) and ready to be textured and shaded. Figure 3.65 illustrates the fit of the output mass-models to the input airborne points to demonstrate how close the output is to accurately representing the input.

3.4.5 Processing Unstructured Terrestrial Datasets

Interestingly, although the MMAMAL algorithm was primarily designed to operate on aerial range-scans, initial speculative experiments demonstrate its surprising suitability for processing unstructured ground scans. Figure 3.66 demonstrates the result of applying the maximal area roof-shape segmentation (MARS) to a high-resolution unstructured terrestrial scan dataset. The total execution time for the scan shown in figure 3.66 was 152.585 seconds (just under three minutes) on a quad-core i7 with 16GB of RAM. Of this 1.737 seconds was spent on algorithmic processing, 3.313 seconds spent writing the output data to disk and the bulk of the time (147.535 seconds) was spent discretising the original input point-cloud (115,694,669 points totalling 6.4GB). Note: that the important aspect of this result is not so much the execution time - but rather the applicability to an arbitrary scan outside the scope of its intended/designated operational remit. In other words this demonstrates that even without explicit support for ground-based scans - MAMMAL is both generalised and versatile enough to adapt to such data.

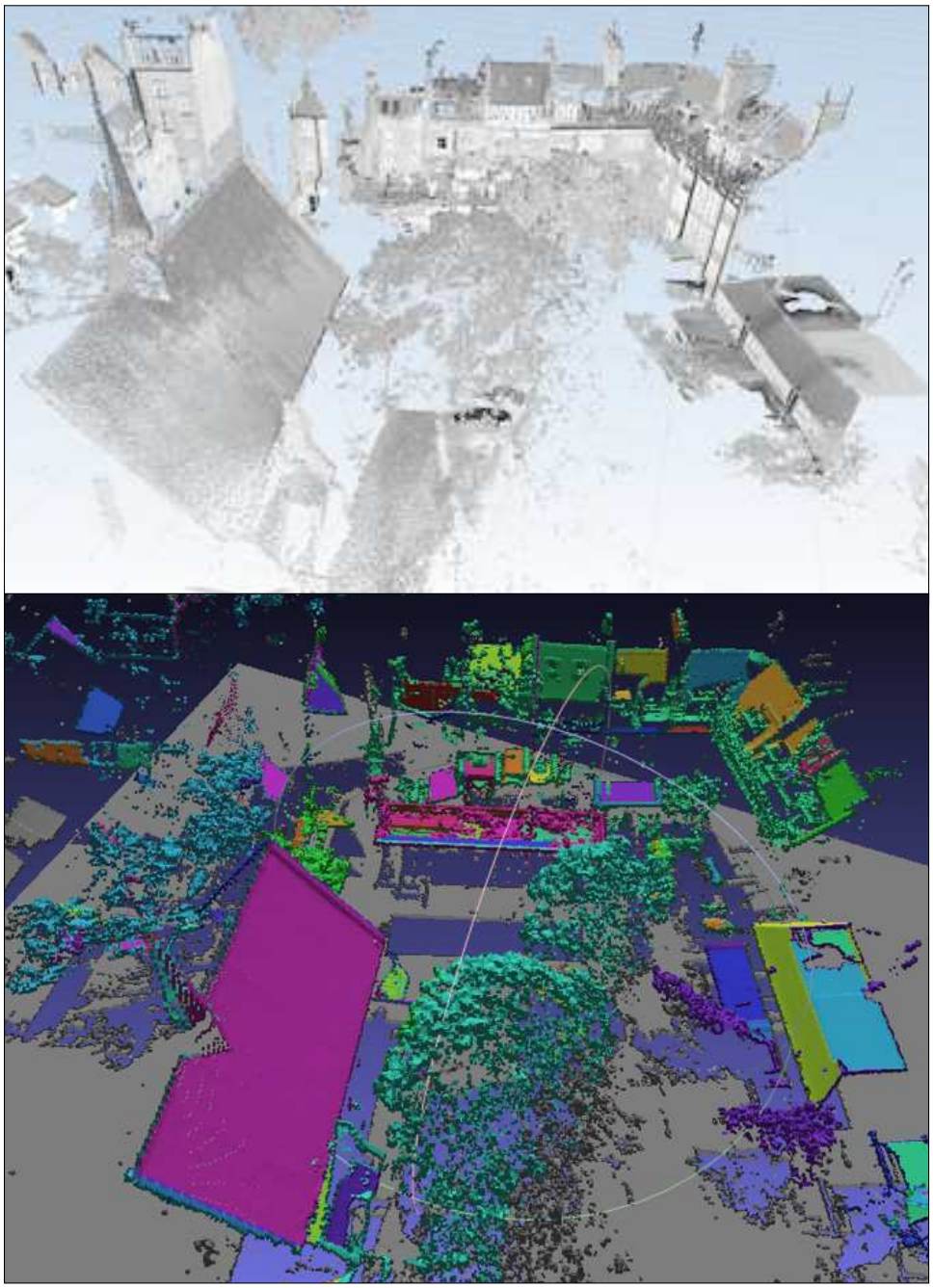

Figure 3.66: *the input (top) and output (down) of maximum-area roof-shape segmentation of a high-resolution unstructured ground-based laser scan*

The scan in figure 3.66 covers buildings in the Roslyn Mews area, that *vitally* also include partial roof points. This is not always the case and depending on the relative location of a scanner during a ground survey roof points may be omitted. In spite of this the response of MARS on this example dataset is promising. Not just because it isolates roof components well, but also in the manner that points corre-

sponding to trees and vegetation are treated (figure 3.66 - scattered tree points assigned to the same regions). This demonstrates some of the behavioural benefits of the object level classifiers and the flexibility of the MAMP principle in its (somewhat surprisingly effective) application to high-density unstructured scan datasets.

Crucially this indicates that the spatial-domain oriented MAMP principle may be applicable to alternative types of scan beyond architectural. As such, from this another interesting area (and a topic for future investigation) is the application of the MAMMAL algorithm's approach to scans from time-of-flight and patterned light sensors. The critical aspect in this regard is replacing the difference-of-elevation models (employed by MARS) with an alternative depth based *splitting* operator.

***

In revision this section documented the results of experimental tests and performance profiling of the airborne building reconstruction algorithm MAMMAL. It included quantitive and qualitative outcomes for city-scale airborne datasets at various resolutions. The next section covers the analysis of these results in order to draw deeper insight into the meaning and significance of these outcomes.

3.5 Analysis and Evaluation

This section deals with analysis and evaluation of MAMMAL's performance. It discusses observations related to the efficacy of each algorithmic stage.

3.5.1 Point-Set Segmentation

The results indicate that MAMMAL's point set segmentation results in efficient data-driven isolation of architectural objects in top-down (airborne) range-scans. The results point to the fact that the MARS strategy is reasonably robust to sensing noise, and that (in particular) the maximisation of the mean roof-shape area - yields higher quality planimetric divisions - which support the subsequent vectorisation process. Interestingly - the segmentation results also indicate that the quality of the roof-shape segmentation masks produced by the algorithm improves as the resolution of the airborne scan increases - which suggests that MARS benefits greatly from denser input scans. However point-spacing has less of an impact on the stability of the DoEM method. This suggests that the most cost effective type of data (from a segmentation perspective) depends upon whether one desires basic contextual models or highly detailed models.

Another important observation from the segmentation results is the fact that non-conformal suppression does not necessarily guarantee that each roof-shape will meet the user supplied minimum area tolerance. Despite this however, the number of non-conformal roof-shapes is shown to be negligible irrespective of the point spacing - which basically ensures that the mean surface areas still exceed the user supplied minimum desired for areas greater than $5m^2$. The results demonstrate that this holds true for scans of upto 25cm point-spacing - though it is important to note as denser airborne scans become available this has the potential to change.

Crucially noise-cancellation and non-conformal suppression of smaller clusters are

vital to reducing the cardinality of each building's set of roof-shapes, and maintaining the desirable maximal-area property. Indeed without them the subsequent processing stages are subject to far greater latency. However it is fair to say that based on the results more research and development is required to improve MARS' current maximisation strategy. In particular - though the performance is quantitatively satisfactory, qualitatively there is still room for improvement.

3.5.2 Point-Set Vectorisation

In evaluating the results of vectorisation, it is clear that scan conversion is a highly useful sub-component of the transition from point-clusters to vector shapes - since it is not only a long-standing technique, but further it is shown to produce effective results efficiently. Essentially rather than invent a new approach to address dense boundary extraction, MAMMAL builds upon a bread and butter computer graphics strategy that can reliably deal with concave and complex polygonal forms.

However the real juice in MAMMAL's vectorisation turned out to be (rather surprisingly) the effectiveness (and indeed the overall take-up/recall) of the freshly proposed shape approximation methods. QUALM and GRAILS represent novel contributions to the problem of 2D boundary extraction that not only perform well in the presence of lower-quality input point-clusters but directly support the goal of producing high-quality compact geometric representations. Indeed based on the results (particularly in the city of Bath) it is fair to say that these approximation functions enable coherent geometry to be recovered even in the presence of imperfect segmentations. In evaluating these aspects of MAMMAL it is quite clear, that the addition of more advanced domain specific shape detectors has possibly the greatest potential to enhance the cleanliness of the results.

However the results also point to the fact that better strategies are required for dealing with vector shapes exhibiting partial curvature. So whilst the oval detector (that feeds the conic-section generator) is useful, in and of itself it is not enough. This is most apparent in evaluating the ALV versus OSM results in the city of Manchester dataset. Note that although the geometric accuracy of the vectorisation of the public library example is satisfactory, ideally it would be quite desirable (from an aesthetic perspective) for MAMMAL to be able to freely mix piecewise planarity with circular arcs at the level of entire buildings and not only individual roof-shapes.

3.5.3 Vector-Shape Projection

Evaluation of the results indicates that MAMMAL's data-driven parallax projection strategy performs poorly on 1m point-spacing airborne scans. This seems to be a product of the fact that there simply are not enough points present to produce coherent projections (in a purely data-driven manner) at that resolution.

On the flip side, the results demonstrate that the linear, and sub-division projections grow more stable as the density of the points increases. What this means is that practically at less than 50cm point spacing, one cannot necessarily rely on the data-driven projections to produce good looking models, and though the resulting building masses may meet the user supplied error tolerance, the geometry returned can only safely be used for analysis and not visually oriented tasks. This also seems to apply to the graph-based roof-shape refinement process (that mitigates height discontinuities between neighbouring projected roof-shapes) - which

also performs better as the resolution of the input point-set increases. However it is important to note that this may also be indicative of the fact that the snap-nodes-and-collapse heuristic used may be insufficient to deal with lower density scans, where the distance between neighbouring nodes (vertices) is correspondingly greater. Further more, the results noted the roughly linear growth in execution time (relative to the number of input points) that is associated with the projection process. Positively this quasi-linear growth in runtime complexity is constant independent of resolution and suggests that in driving the transition from 2D to 2.5D without stochastic sampling significant performance gains can be achieved relative to pre-existing methods such as [73] and [142].

3.5.4 Non-Linear Procedural Optimisation

In considering the results of MAMMAL's procedural optimisation it is evident that the functionally driven strategy is generally more efficient than traditional library-based model optimisation approaches that rely upon analytic solvers and stochastic sampling. A vital difference is that MAMMAL's optimisation kernel amalgamates the desirable features of both model-based and data-driven techniques (through the use of open and fixed-form generative parametric functions) and further more reformulates the problem of high-quality model recovery such that the process of iterative sampling is addressed in a non-stochastic manner.

Ultimately the procedurally optimised building models produced by MAMMAL typically exhibit higher geometric and topological quality than the purely data-driven projections, however generally the projected models will surpass the parameterisations in terms of geometric accuracy. Interestingly the results indicate that the use of sequential parameter variation mitigates the massive combinatorial (exponential) explosion in execution time that would otherwise be witnessed as a product of an increasing number of arguments to the fixed-form functions without overly detrimental restrictions. This means that although (in the strictest sense) the traversal strategy is not truly exhaustive - this does not constrain the versatility of the kernel.

Positively the results demonstrate the benefit of exploiting the CSG boolean union operation - since it trivialises the process of transitioning from sets of the parameterically optimised masses to single watertight building shells. However practically it proved necessary to employ a sub-optimal BSP re-massing operator over an optimal (topology-preserving) variant in order not to bloat the execution time of the final stage. This aspect could be improved. By considering the results it is clear that the crucial attribute of MAMMAL's optimisation kernel (and indeed its utility) is the succinctness and semantic richness of the building representations produced.

One thing that is apparent is that this entire ideology rests upon the fast parallax rasterised depth buffer error - without which this functional approach simply would not be feasible or practical. This is problematic because ideally - one would like the ability to extend this approach to full blown 3D datasets - however at present this is not immediately possible whilst maintaining the same performance traits.

Another evaluative observation drawn from the results relates to the tiny set of experimental functions that are currently employed by MAMMAL. Essentially there is further scope for extending the versatility of the method by adding generative functions. Nonetheless the optimisation strategy employed represents probably the most significant contribution that MAMMAL embodies - because it opens the

doors (lays the foundation) for the recovery of smart building model components.

3.5.5 Computational Efficiency

The results demonstrate that the approach is surprisingly fast relative to the pre-existing sparse reconstruction methods. This is particularly evident when one compares the runtimes of MAMMAL to that of the current state of the art in high-quality airborne massing model recovery [73], [167], [75].

Without overstating how important this is - MAMMAL effectively does in seconds that which it takes existing algorithms minutes to achieve - and in minutes that which currently takes hours to do. This signifies a massive leap forwards in-terms of reducing the turnaround time for high-quality architectural reconstruction from airborne laser-scans. MAMMAL is in effect an order of magnitude more efficient than many highly regarded methods. However beyond this the real power lies in the fact that the current implementation does not even represent an optimal solution. Indeed the improvements and enhancements section discusses numerous mechanisms and strategies for further reducing MAMMAL's computation times.

To put this into practical terms (that may be of use to a technician) - at 1m point spacing MAMMAL reconstructs 1km^2 (1,000,000 points) in less than 5 seconds. At 50cm point spacing MAMMAL reconstructs 1km^2 (4,000,000 points) in less than 30 seconds. At 25cm point spacing MAMMAL reconstructs 1km^2 (16,000,000 points) in less than 100 seconds. What this means in that given an urban region covering 10km^2 (the area of an OSGB36 tile sub-cell) MAMMAL is capable of extracting building models in less than 10,000 seconds (100 seconds x 10km x 10km $\approx$ 166.6 minutes $\approx$ 2 hours and 45 minutes). Based on this (conservative) over-estimate - the whole of the United Kingdom - which (again as an over-estimate) covers roughly 50 100km by 100km OSGB36 tiles - could be reconstructed in less than 50,000,000 seconds (or 50 x 100km x 100km x 100 seconds) which is just under 14,000 hours (or 580 days) on a single commodity machine using 25cm point-spacing airborne LiDAR. Now if one reduces the resolution to 50cm point-spacing the entirety of the UK could be reconstructed in just under 175 days (on a single machine) - and at 1m point-spacing this becomes just under 45 days. Note that all of these figures aggressively over-estimate the time it takes MAMMAL to reconstruct airborne massing models in order to determine the maximum upper-bound on the total execution time. However in practice large parts of the UK will be sub-urban and rural and hence the building reconstruction times for such areas will be considerably less. Further more the really interesting aspect of this forecast is that it assumes a single commodity machine is employed - i.e. an independent researcher/engineer churning out architectural models on their laptop or desktop computer. Practically however things really heat up when a suitable parallelism strategy is employed alongside hardware for distributed computing. For example by employing a homogenous cluster of (as an example) 16 such commodity machines one can reduce the overall reconstruction time of the UK to 1/16th of the values quoted. With 50 such commodity machines - 1/50th of the time quoted would be required - and so on and so forth. Essentially it should be quite obvious that the linear growth in runtime of the MAMMAL algorithm, alongside the data independance effectively supports the simultaneous reconstruction of country scale airborne laser scans. In this sense it is fair to conclude that MAMMAL meets the efficiency and scalability objectives set out at the beginning of this chapter.

3.5.6 Qualitative Analysis

The building models produced by the MAMMAL algorithm are qualitatively satisfactory. At this stage that is all that can be said. In relation to the objectives set out in this chapters overview - the models produced are sparse enough to be rendered interactively and reasonably close in appearance to the sort of massing models that would be produced by a human CAD technician. However it is important to be aware that this is a subjective judgement based on the goals of this project's industrial sponsor. It is not a definitive statement regarding the absolute quality of the geometry produced. Ultimately the more important attributes are MAMMAL's efficiency, accuracy and sparsity - because they can be clearly quantified. Nevertheless - given that low-quality building models were cited as one of the key limitations of the pre-existing airborne reconstruction methods - it is important to give some consideration to the visual quality of the geometry produced.

So at a high-level, it is fair to say that the models look good (they are aesthetically pleasing to an extent and immediately recognisable as buildings). However this does not mean they could not look even better. Interestingly it seems that the *visual-quality* of the results are tightly coupled to the scale of features that are represented - and balancing the preservation of roof-components with the desire for clean looking geometry can be quite difficult. However to give the reader a clearer sense of how this manifests, figures 3.67 and 3.68 illustrate instances of LOD1 and LOD2 automatic massing models in order to complement this discussion.

One key thing to note in these figures is the visual appearance of the example city of Manchester public library (top-left for LOD1 and top-row, second from the left for LOD2). Note that although the LOD1 model is clearly less accurate than the LOD2 model - in many respects one could argue that it is qualitatively a better representation since it looks slightly more regularised as a product of the symmetry of the oval and quad mass components. These figures epitomise the current qualitative state of the building models generated by MAMMAL.

Figure 3.67: *architectural mass-models reconstructed from the city of Manchester (25cm point-spacing) and city of London (50cm point-spacing) datasets - illustrating results of the data-driven projection and parametric optimisation applied by MAMMAL.*

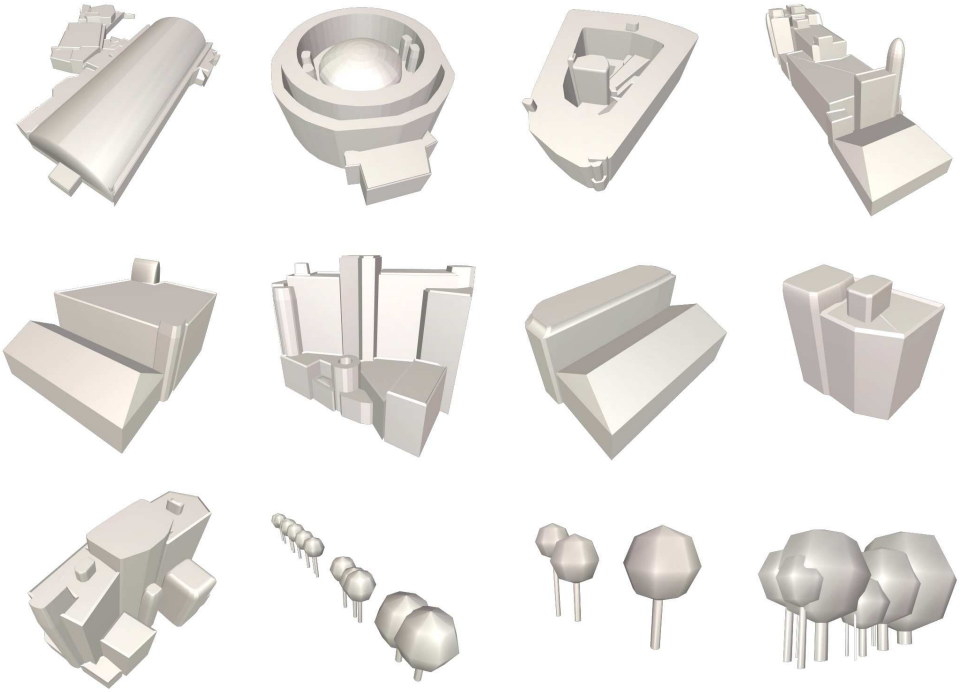

Figure 3.68: *further mass-models reconstructed from the city of Manchester dataset (at 25cm point-spacing) - illustrating results for a selection of distinctive buildings approximating LOD2 representations and simple tree models produced automatically by mining maximally inscribed discs from point-clusters classified as corresponding to organic vegetation.*

3.6 Enhancements and Improvements

This chapter presented MAMMAL (that is Maximal Area 2.5D Mass Modelling of Airborne LiDAR) which is a fresh approach to tackling a long-stand reconstruction problem. MAMMAL aims to maximise the mean surface area of polygonal faces in output 2.5D building mass-models so as to ensure sparse, high-quality architectural geometry is generated. Furthermore MAMMAL employs a generalised and expandable procedural approach to model optimisation based on efficiently identifying best fitting arguments for user supplied generative modelling functions - and a fast rasterised depth-buffer error measure. These features make the MAMMAL algorithm highly adept at constructing interaction ready compact building models in a manner that is resolution independent and scalable.

However the current implementation of MAMMAL is not perfect, and (given more time) there are a number of aspects that the author would refactor and/or improve so as to enhance MAMMAL's performance. This newly introduced section outlines these future investigative tracks. The improvements are organised according to the attribute that each seeks to address - with the fundamental categories being, geometric accuracy, computational efficiency, geometric quality and robustness.

The underlying aim of this section is to clearly document the aspects of the current implementation of MAMMAL that still require additional work so as to help and guide future researchers charged with generating sparse architectural geometry from airborne laser scans. This is in essence a road map of the remaining problems that still require addressing further - that one might choose to focus upon.

3.6.1 Geometric Accuracy

The following enhancements and improvements seek to reduce the geometric error of the 2.5D mass models generated by the MAMMAL algorithm.

- Constrained Vertex-Edge-Face Offsetting - Post Reconstruction Registration. This involves changing the position of vertices in each output mass model by sampling (*jittering*) new potential positions (in the locale of each vertex) and considering whether each offset vertex position reduces the model's overall error. This would effectively decrease the associated error for each model, however it would also alter any regularity or uniformity embedded in the optimised masses. This would mean for example that although the mass models are more closely aligned to each segmented point-set, practically features such as right-angles and parallelism between faces could be lost.
- Super-Sampled Representations. The use of multi-resolution representations in order to increase (and decrease) the density of each airborne scan so as to enhance the stability of the projection from 2D to 3D. This could be as simple as exploiting range-image-pyramids. However more complex (non-linear filtering) strategies could aim to prioritise the up-sampling of roof-shape points over near vertical wall points.
- Recursive Application of Sub-Division Projection Routines. Instead of applying one level of subdivision to each footprint or roof-shape, the idea here is to consider the error of each subdivision and (rather than simply decreasing the desired surface area of sub-clips to increase the level of detail) identify

sub-clips that fail to meet the error tolerance and re-apply the polygonal-subdivide in order to construct *nested* subdivisions which exploit two or more distinct splitting functions.

Whilst there are a myriad of ways to reduce the geometric error associated with each mass-model, practically these three enhancements represent the prominent generalised approaches. Since geometric error is one of the principal evaluative measures these also represent high-priority changes.

3.6.2 Computational Efficiency

The following enhancements and improvements seek to reduce the execution time and memory requirements for the generation of 2.5D mass models.

- Parametric Optimisation Boosting Methods. In particular the approaches to improving the runtime response of the generative modelling functions employed during non-linear optimisation. These largely boil down to two ideas. First: reducing the number of functions that are traversed for each building and, secondly: reducing the time taken to evaluate a function.
- Parallelism and Distributed Processing. As a means to spread the execution of MAMMAL over simultaneous architectures. The nature of the algorithm means that each segmented building can be reconstructed in isolation. This can be coordinated in a number of ways - with the simplest being multi-core (CPU-bound parallelism). A faster approach would be to GPU parallelism.
- Hardware Based Rasterised Depth Buffer. Transitioning to a lower-level depth buffer instead of the software based variant that is currently used. The key benefit of this is significantly reducing the time it takes to evaluate candidate mass-models. However the key downside is that MAMMAL would no longer be platform-independent - because different builds would be required for each architecture supported.

Although reducing execution time and memory use is important in terms of MAMMAL's scalability, it should be noted that even without these proposed further advancements MAMMAL still executes substantially faster than the current state of the art high-quality mass-model reconstruction algorithms. The critical point is that whilst speed is crucial, the pursuit of it should be secondary to the pursuit of model accuracy and quality. The difficultly is that oddly speed (i.e. an algorithm that executes quickly) will generally be easier to improve since a greater number of development (refactor, test) cycles can be coordinated within a fixed period.

3.6.3 Geometric Quality

The following enhancements and improvements seek to further improve the semantic and structural quality of the 2.5D mass models recovered by the MAMMAL algorithm. Note that unlike the alterations targeted at enhancing computational efficiency and reducing geometric error, these alterations represent qualitative improvements which (though to an extent subjective) aim to change the aesthetic properties of the geometric models. In effect these alterations will help the architectural models look better, but will not necessarily reduce the associated error

or the time taken to converge. This is an important distinction and as such the benefits of these changes should be carefully considered relative to their expense.

- Topology-Aware Segmentation. In terms of optimising the arrangement of roof-shapes earlier on in the reconstructive process as part of the area maximisation (non-conformal suppression) stage. This involves embedding additional constraints on the formation of regions into MARS. Positively the use of additional priors could help deal with low quality segmentations. Negatively this could (to an extent - depending on how it is implemented) negate the data-driven nature of the segmentation - and introduce greater error.
- Explicit Component-Level Curvature Detection. As opposed to the error-based sub-division and parametric approaches to representing curved roof-shapes currently employed. The core idea is to determine (before polygonisation) which parts of a building map to curved and irregular features and use this information to alter the strategy employed for these parts. The critical benefit is that knowledge of parts that correspond to conic sections (for example) would enable MAMMAL to effectively force higher quality approximations for such parts using functions such as the radial rail. The crucial pre-requisite to this however is robust curvature detection, which represents a long-standing (but addressable) scale space problem.
- Piecewise Intersection of Planes. As an alternative linear projection refinement technique, specifically for primarily planar roofs.
- Enforcing Regularity Constraints. This involves altering the layout of the vectorised shapes, without butchering (mis-characterising) the resultant geometry. The challenge lies in this being an ill-defined problem. Though strategies such as edge-length maximisation work well at the component level, the thing that is lacking is strategies based on inter-component relationships. Further while regularity can improve the visual quality of a model, its enforcement invariably comes at the cost of the fidelity of the approximation. Commonly researchers have addressed this with the use of *snap-line* techniques [166] and considering principal directions [148] - however there remains a lack of a definitive formalism. Still this is an alteration with massive potential to increase the quality of the building models MAMMAL generates.
- Alternative Graph-Theoretic Refinement Routines. In order to complement the *cluster→collapse* method currently used to simplify roof-shape nets.
- Dealing with Partial-Scans. In particular the inclusion of hole filling methods that can be effectively applied to range-scans to address regions of missing-data within the boundaries of buildings.

3.6.4 Robustness and Stability

The following enhancements and improvements seek to make the MAMMAL algorithm more robust to geometric degeneracy and undesirable sensing artefacts. Note: these alterations will generally not impact the large majority of building instances, however they will ensure that the minority of edge cases, that may sometimes result in erroneous results are correctly handled.

- Arbitrary-Precision Arithmetic. At present fixed precision arithmetic is used in the implementation. By and large (for the image-based operations such as segmentation, scan-conversion and graph-refinement) this is not an issue. However for operations such as the polygonal sub-division routines and the variant of the BSP algorithm (used in CSG re-massing), fixed-precision arithmetic can be critical - since it can lead to degeneracy or a non-manifold return. The obvious choice to combat this is CGAL - however it is quite heavy-weight, adding a significant dependancy to MAMMAL. There are also implications in terms of licensing for industrial use of CGAL. The simple alternative is to implement one's own arbitrary precision versions of each of the algorithms riddled by numerical precision issues.

There are in effect a number of advancements that can be made to MAMMAL. The critical challenge is to balance the speed and accuracy of the revised reconstruction methods with the mandate for compact, high-quality returns. The next section concludes the airborne reconstruction research.

3.7 Discussion and Summary

In conclusion, this chapter presented a simple yet effective algorithm for reconstructing compact 2.5D building mass models from airborne laser-scans in a highly efficient manner. The behavioural properties of the algorithm make it quite useful when compared to the currently employed techniques. It is simultaneously fast, accurate and sparse! The chapter detailed the four key processing stages (segmentation, vectorisation, projection and optimisation) and considered the vital factors that must be controlled in order to maintain efficient performance. The chapter also presented and analysed experimental results of performance profiling with three different city-scale aerial laser scan datasets (at varying point-spacings), documenting for each the quantitative and qualitative response of MAMMAL - and for the City-of-Manchester - comparative results relative to pre-existing algorithms.

Whilst the results are promising (and indeed one could argue superior to existing sparse reconstructive methods in many regards), there remain many things that could be even better. The newly introduced enhancements and improvements section outlined these key areas for further investigation. Each future research track aims to enhance one of the performance attributes: geometric accuracy, computational efficiency, geometric quality and robustness.

Further an important observation drawn from these experiments is the necessity for equivalently sparse facade models. In essence although MAMMAL serves its purpose well, it leads naturally to the requirement for accurate window and door models to reside on the walls of reconstructed massing models. Although in certain situations (such as sky-line rendering and the production of distance still-images) one could argue that *surface-element* models are simply overkill, for most interactive simulation purposes, their omission limits the scope of that which may be simulated. Given that buildings are characterised as being the union of mass and surface [22], in recovering massing models, it also becomes necessary to recover surface-element models as without them a building's representation is incomplete.

Finally, for reference, the key ideological concepts covered in this chapter, as well as the novel technical developments and the auxiliary factors critical to ensuring efficient algorithmic performance are revised below:

- **Maximal-Area : Minimal Primitives (under an error tolerance) Principle** as a fundamental logical paradigm for fast, accurate and sparse geometric reconstruction - since in minimising the number of objects to be polygonised each result can be computed more efficiently (given that there are fewer objects to evaluate) whilst also enhancing the compression properties (since the algorithm will yield the lightest representation it can find that conforms to a given error threshold).
- **Determinism (Our Salvation, Our Joy)** as a fundamental pre-requisite to procedurally controlling the reconstruction of airborne massing models.
- **Maximal-Area Roof-Shape Segmentation** in order to produce clean higher-quality segmentation masks than conventional data-driven roof segmentation methods - as a product of actively re-arranging the result so as to (analogously) minimise the number of salient regions and further more remove sources of latency in subsequent processing stages.

- **Continuous Scale-Space Level of Detail** controlled by the user-supplied input minimum desired roof-shape area (in meters^2) - which enables the dynamic masking (suppression) of features at variable spatial scales.
- **Graph Refinement Operator for Vector Extraction** a 2D raster to vector converter that preserves the topology of boundaries between neighbouring regions by anchoring vector simplification about convergence points (which are defined as being adjacent to 3 or more regions).
- **Quick-Unconstrained Approximate L-Shape Method** an intuitive domain specific 2D building footprint boundary extraction algorithm for data-driven recovery of high-quality *eat-away* L, T and S shapes.
- **Graph Refined Approximate Interior Linear Spine** a spine detection operator (a specialised form of skeletonisation) that was designed to support automatic terrace model reconstruction by extracting maximal length non-cyclical piece-wise linear paths from bidirectional graphs. Each intermediary graph is constructed discretely from the loci of the maximally inscribed discs of input 2D polygonal boundaries.
- **Structured Sub-Divisions for Non-Linear Roof-Surface Projection** in order to control the distribution of vertices over roofs in a data-driven manner that vitally, can approximate curvature and planarity directly without the requirement to first resolve semantic structure.
- **Data-Driven and Model-Based Procedural Optimisation Kernel** to marry the benefits of exploiting both fixed-form and open generative modelling functions for automatic enhancement of 2.5D massing models recovered from airborne laser-scans - in a manner that is direct, generalised and extendable. A by product of this strategy is the resolution of compact implicit parametric representations that can be interactively edited with semantic controls.
- **Rasterised Depth-Buffer Error-Measure for Parallax Optimisation** crucial to enabling efficient functionally based optimisation of 2.5D building masses. Without a rapid means of measuring the geometric error between airborne points and candidate mass-models the strategy proposed for procedural reconstruction would not be feasible.
- **Spatially Aware BSP Algorithm for Constructive Solid Re-Massing** a simple extension to a long-standing CSG algorithm that seeks to preserve as much of the original topology of parametric masses as possible during the union operation by effectively un-doing unnecessary clips.
- **Algorithmic Decision Making via Model Selection Criteria** as a generalised means of objective-driven traversal of (and ranking of instances in) the space of potential mass-models suited to reconstructing a building. The model-selection criteria enable an end user to specify a high-level guiding desire that controls the manner in which a scene is recovered, that vitally is abstracted from the particulars of the domain.

The next chapter progresses this research - tackling the problem of automatic facade model reconstruction from unstructured ground based laser-scans.

Chapter 4

Ground Facade Reconstruction

What is it?

An operator for segmenting and modelling windows, doors and other rail-able (sweep-able) objects in unstructured ground laser-scans of building facades.

Why does it exist?

To enhance the accuracy, efficiency and quality of data-driven facade modelling from unstructured point-sets - enabling analytic tasks such as automatic building energy analysis and greater physical precision in simulated scenes.

How does it work?

By a two-stage segmentation that isolates salient clusters of points on a facade, followed by skeletonization using a constrained transport-rail detector.

surface detail...

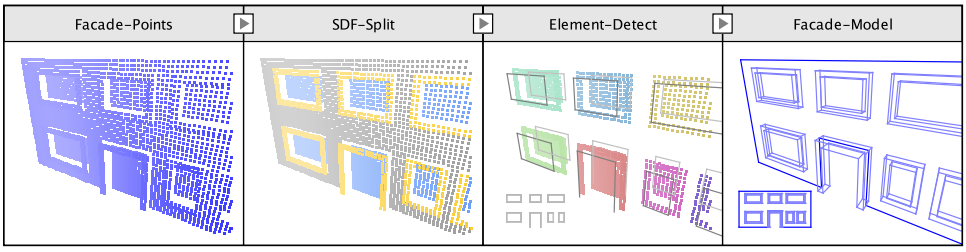

Figure 4.1: an overview of the key stages in ground facade reconstruction - from left to right: (blue) an input ground point-cloud, (blue, gray, goldenrod) signed distance field split, (hsb) segmented and refined disjoint connected-components, the resultant facade model.

4.1 Overview

This chapter addresses the problem of recovering sparse building facade models from unstructured ground laser-scans. The chapter proposes a simple data-driven method of modelling facades that is based on the use of sets of generalised cylinders and a quad-dominant meshing strategy. The aim of the operator is to reconstruct semantically-rich architectural surface geometry in an efficient manner. The context of the chapter is 3D architectural simulation and visualisation of the physical world, in which accurate facade geometry is required to create first-person interactive virtual environments.

One of the key limitations of the pre-existing methodologies is the reliance on dense surface reconstruction. Further more computational efficiency represents a key bottle-neck in the pre-existing facade reconstruction methods - with many operators requiring hours to process high-resolution laser-scans. Additionally - note that whilst researchers have integrated image-based (photogrammetric) techniques to negate some of the limitations of facade reconstruction from laser-scans, the critical problem with such approaches is the reduction in geometric accuracy [70]. Typically in techniques that exploit image data, the objective is to produce aesthetically pleasing facade models [93] - which although suitable for rendering and visualisation, cannot necessarily be relied upon for building analysis [77].

Ultimately the desire is for an automatic facade-reconstruction method that is simultaneously fast, accurate and capable of producing sparse geometry.

The techniques presented build upon those covered in the previous chapter - however now the focus is recovering geometry from unstructured laser-scans as opposed to structured range-images. This represents the step from 2.5D into full 3D and as such the methods defined are also responsible for efficiently imposing the necessary structure upon un-ordered sets of points.

This chapter also introduces a generative modelling primitive for the definition and data-driven construction of high-quality window and door geometry. The *Surface-Element* (as it is termed) provides a layer of abstraction between the processes of analysing facade scans and instantiating geometry - that is well suited to the both automatic reconstruction and user-centric interactive modelling. The *Surface-Element* is an integral intermediary representation that plays a vital role in the principal algorithm defined in this chapter - ARROW (Accurate Railed Reconstruction of Openings and Walls).

As such a considerable amount of attention is devoted to its exposition.

To further clarify examples of concrete use-cases for the facade-reconstruction algorithm defined in this chapter, one could conceive using it:

1. As a window detection operator in order to support automatic building energy analysis in urban environments - for example by computing the solar-gain properties or estimating energy-loss for buildings - by considering the count and surface area of windows and doors.
2. As a sparse reconstruction method for unstructured laser-scans of building facades - capable of yielding semantically rich, component-based models for which multiple material shaders can be applied automatically - whilst also being simultaneously compact enough to be rendered interactively at city-scale within 3D virtual environments.

3. As a generic formulism for the construction of geometry of architectural surface-elements during generative and procedural modelling.

For reference the key objectives, requirements and desired behavioural characteristics of the operator are stated following. Remember this is the means of turning unstructured ground point clouds into clean architectural facade models suitable for interactive simulation and building analysis.

- Geometric Accuracy (correctness/precise/error-bounded results) such that each reconstructed facade model can be used for analytic tasks and to address surveying problems. It is essential that the deviance between the physical position of each wall and the geometry of the corresponding generated models be quantified (and minimal).
- Computational Efficiency - in order to limit the exponential growth in runtime (as a product of the number of input points) and ensure large unstructured sets can be reconstructed on commodity hardware (without the requirement for distributed architectures). Unlike aerial reconstruction - the key challenge here is not simply limiting the growth in execution time, but additionally rigidly controlling memory allocation.
- Deterministic (Non-Stochastic) - such that equivalent facade models can be repeatedly constructed from equivalent inputs (data + control arguments) and to simplify the process of performance analysis. The key idea (and desire) is that the operator should be a reliable (dependable) solution, that simply works out of the box - without the need for iterative stochastic refinement. This property aims to ensure that the technique can be easily integrated (as a stand-alone closed-form plug-in solution) within CAD packages and 3D modelling environments such as AutoCAD, Modo and Rhino. Additionally it eases enhancement.
- Data-Driven - such that irregular surface-elements can be effectively reconstructed directly - without the requirement to template fit - and to ensure the semantic correctness of each facade model. In essence the operator should not look for an approximate. It should model the data-present. This behavioural property is vital, not only for scalability, but to ensure that on close-up inspection each model is visually correct.
- Robust to Geometric Degeneracy and Sensing Noise - especially since this algorithm operates on unstructured point-cloud data, it must be capable of performing effective facade-reconstruction in the presence of variable sampling density data, partial or missing data and even unfiltered input data. This also includes handling artefacts resulting from scanning transparent and reflective surfaces (such as glass windows).
- Efficacy - Competence : in window and door detection - high precision and recall relative to ground-truth data from human operators - which simply means that the position and extents of each detected surface-element should be roughly the same as a manually constructed facade model that a human would create given the same input.
- Sparse with High Visual Quality - in order to enable the generated models to be exploited within city-scale 3D virtual environments

- Semantically-Rich : Component-Driven - such that each reconstructed facade models embodies the structural composition of the actual facade. This enables an end-user to manipulate the result at the level of individual surface elements rather that as an un-ordered mesh.
- Intuitive (to understand and implement) - since (like the semantic change detector presented in the chapter two) this represents a novel (unknown) strategy that serves a niche in geometry processing. Unlike aerial massing reconstruction (which is a particularly high-profile problem), fewer organisations and researchers explore facade-reconstruction. Hence if the method proposed is so complex that few can actually implement it - it is far less likely to be exploited in other domains outside of architecture - and would be limited to practitioners with significant need of automatic window and door geometry. Although algorithmic performance will (largely) be the deciding factor in terms of the algorithms industrial use, the expectation is that a simple, readily understood method is more likely to be integrated into environments such as game-engines, GIS frameworks and 3D mapping solutions. Further such a method will be easier to extend and build-upon in the future.
- Tractable - such that the intermediary data and algorithmic decision making can be trivially intuited by a human technician upon inspection. This is critical, because unlike aerial reconstruction which is reasonably well-defined, there is a lot of scope for ambiguity in the window and door detection process. As such strategies that mask the process of segmentation or overtly obscure the polygonisation stage should be omitted. This characteristic seeks to simplify debugging.
- Capable of Modelling Curved and Irregular Surface-Elements - in order that the data-driven aspects of the window and door element routines are not constrained to rectilinear shape arrangements. This aims to ensure unique ecclesial surface elements (such as the Gothic architecture of the Cologne Cathedral) can be effectively reconstructed. This is the vital behavioural property that distinguishes this method from template-based strategies - and indeed provides the greatest leap forwards towards true data-driven automatic facade model recovery.

Whilst this list of objectives may seem extreme, these are the key attributes required to combat the limitations of the existing methods and represent the behavioural characteristics necessary to ensure the method proposed is both generalised enough to deal with the architectural diversity present in the real-world and efficient enough to be exploited at city scale. Ultimately the key benefit of pursuing an operator that possesses these attributes (and to a large extent the novelty of this approach) is the ability to produce accurate, high-quality facade geometry in an efficient non-stochastic manner.

The remainder of this chapter is structured as follows:

- The background and context section details the vital pre-existing techniques to automatic facade reconstruction. It covers the common methods used to address the problems of segmentation, polygonisation and surface-element recovery, enumerating for each the benefits and limitations. It seeks to provide a concise synopsis of the strategies that informed and inspired the proposed ARROW algorithm.

- The methodology section defines the ARROW algorithm in terms of the key processing stages exploited (which are slicing, dicing, railing and clipping). It provides a complete overview of the method (to begin with) and then delves into the meat of the method - in a similar manner to the previous two chapters.
- The results section documents outcomes of performance analysis using several unstructured laser-scans of urban facades. The section considers both quantitative and qualitative measures of the techniques behaviour, and discusses the results of window detection (in terms of precision and recall) relative to results created by 28 human operators.
- The analysis and evaluation section details the high-level analysis of the ground facade reconstruction operator. The section examines and expands upon the results in order to draw additional insights from the experiments. The aim is to present a structured break-down of the reasons why (and aspects for which) ARROW is successful and expose the critical limitations that can degrade its performance.
- The improvements and enhancements section follows the pattern of the previous chapter and discusses the topics and areas for further investigation. As before this section is structured according to the performance attributes that each future optimisation addresses.
- The discussions and summary section provides a synopsis of this chapter - revising the aims and outcomes and commenting on the implications moving forward. It seeks to be concise and uses bullet-point notation to reiterate the key ideologies, formal-abstractions and technical developments presented in the body of the chapter.

4.2 Background and Context

Before discussing ARROW's methodology - the relevant prior research is briefly recapped. The aim is to revise the alternative approaches to facade reconstruction.

4.2.1 Key Related Work

The section categorises the facade reconstruction methods discussed based on the type of data they operate on. It covers LiDAR-based (laser-scanning) methods, image-based (photogrammetric) methods and hybrid methods.

LiDAR-Based Facade Reconstruction

There are four key prior contributions to this area that inform aspects of this research. Firstly the automatic facade segmentation method of Martinez, Soria-Medina, Arias and Buffara-Antunes [77] - which produces accurate facade decompositions in a data-driven manner based on statistical layered analysis. Secondly the automatic segmentation strategy of Hao, Wang, Ning et al. [50] who rely on segmenting planes and evaluating Gaussian images. Although both have their merits they primarily address the segmentation problem rather than modeling.

The third approach by Lin, Gao, Zhou, Lu et al. [75] does address modelling (as

well as segmentation) however it does not represent windows and doors. Despite the input being terrestrial scans their primary aim is mass reconstruction taking into account non-parallax features (i.e. things that are occluded in airborne scans).

The fourth strategy (of Truong-Hong, Laefer, Hinks and Carr [141]) does produce models that represent apertures accurately - however it is intended to support building-energy efficiency simulations (i.e. FEM analysis) rather than city scale reconstruction. The angle-criterion and voxelisation work well but the heavy-weight conformal mesh that are generated do not support component-level revision.

Image-Based Facade Reconstruction

Conversely - many of the image-based facade modeling operators do support construction of compact component based representations - for example the early influential work of Müller, Zeng, Wonka and Van Gool [93] and Xiao, Fang, Tan et al. [153]. However both rely on rectilinear (axis-aligned) arrangements.

This is a common theme in image-based operators - and though more recent techniques support more general arrangements [110] they typically still represent detected features with bounding-boxes. This includes the work of Teboul, Simon, Koutsourakis and Paragior [139], Kulkarni, Nagesh and Wu [63], Martinovic, Mathias, Weissenberg and Van Gool [79] and Miljanovic, Eiter and Egly [89]

More recently Martinovic and Van Gool [80] use Bayesian methods to learn facade grammars. Wu, Yan, Dong, Zhang and Wonka [152] also tend to inverse procedural modeling of facade layouts, whilst Cohen, Schwing and Pollefeys [21] use dynamic programming to improve facade grammar parsing efficiency.

Despite the compact descriptors that image-based operators typically yield - the key limitation of these approaches is the reduction in accuracy/fidelity - relative to LiDAR-based methods. They are great for visually oriented tasks but cannot necessarily be relied upon for surveying and analytic tasks.

Hybrid Facade Reconstruction

The hybrid approaches operate on both 2D and 3D representations. Some use MVS and SfM derived depth data - for example: Wu, Agarwal, Curless and Seitz [151] Lafarge, Keriven, Bredif and Vu [69][70] and Martinovic, Knopp, Riemenschneider and Van Gool [78]. Others rely on sparse geo-referenced point samples - such as Bacharidis, Sarri, Paravolidakis et al. [5]. Others - such as Hohmann, Krispel, Havemann and Fellner [51] and Pu and Vosselman [105] have also proposed hybrid systems that operate on actively sensed point clouds and images.

Though they aim to captilise on the complementary strengths of images and points - they are all subject to the same additional reconstructive challenges: multi-modal registration and non-commutative inference between distinct sensing mediums.

To recap - there are comparatively few LiDAR-based facade reconstruction operators relative to the prevalence of image-based and hybrid methods. This can largely be attributed to ease of access to data and the technical challenges inherent to processing unstructured point-sets relative to 2D representations.

To advance the state of the art (in this area) the key objective is to produce facade representations that are faithful to the data yet compact and component-based.

4.3 Methodology

The algorithm defined in this chapter is designed to efficiently and accurately reconstruct sparse, light-weight architectural facade-models from unstructured ground based laser scans of buildings. The ARROW algorithm is a data-driven modelling strategy that seeks to marry the benefits of explicit methods (computational efficiency) with the topological quality associated with template-based (or library-driven) methods, without compromising the integrity (fidelity/precision/correctness) of the resultant 3D models.

Like the developments of the last two chapters the key algorithmic steps can be broken down into a number of independent processing stages.

Figure 4.2: *Key processing stages of the **ARROW** algorithm:*
→ Accurate Railed Reconstruction of Openings and Walls

Figure 4.2 provides an indication of the algorithmic steps which involve:

1. Slicing the input facade-scan points in order to yield a binary division composed of one set for wall-points and one for salient interest-points.
2. Dicing the set of salient interest-points into distinct connected components by considering the disjointness of clusters of points.
3. Railing each segmented connected-component using swept profiles in order to create 3D window and door models from clusters of points.
4. Clipping a polygonised version of the set of wall-points using the boundaries of each connected component in order to create a *cut-out* wall mesh with holes in place for each corresponding window model.

This portion of the chapter delves into the logic of each of these stages, outlining for each the vital considerations and the algorithmic steps that implement them. A complete overview of the approach is also included, which is followed by the discussions of the slice, dice, rail and clip in order.

4.3.1 Outline

This section provides a provides a brief outline of the structure of the ARROW algorithm. It explains the input and output data as well as providing pseudo-code to clarify the lower-level behaviour details. There are a number of extra figures included in order to drive home the algorithm's behaviour.

Inputs-Outputs

The facade-reconstruction algorithm takes as input: an unstructured .pts or .xyz point cloud file - a collection of points scanned from the surface of a building (typically using a fixed-position radial scanning setup).

The ARROW algorithm makes use of the following control-parameters (which are

supplied as input-arguments by an end-user): an iso-distance (a signed scalar value), a point neighbourhood radius (an unsigned scalar value) and a quadrilateral edge-length (an unsigned scalar value in centimetres).

The output is a 3D facade model which is composed of a quad-dominant wall mesh and a set of surface-element meshes each representing a window or door. Each surface-element model is composed of a set of 3D sweeps which are generated as the product of a 2D split-logic descriptor derived from the window's/door's interior sash and pane arrangement.

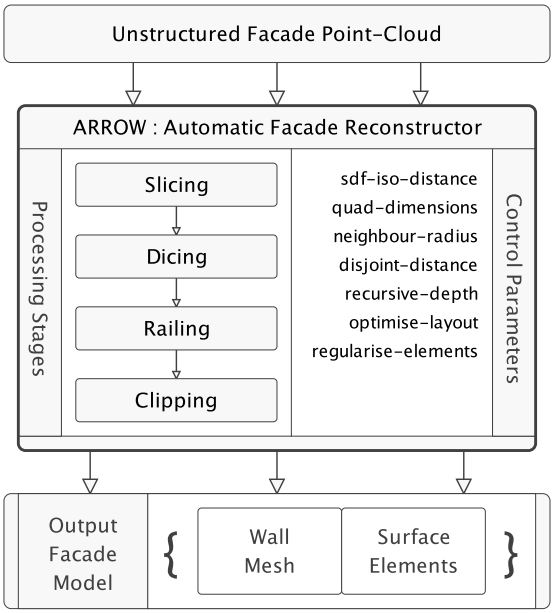

Figure 4.3: *Schematic high-level overview of ARROW- the top row indicates the input data (unstructured facade point-cloud), the central block outlines the body of the algorithm, and the last row the output facade-model.*

Figure 4.3 outlines the structure of the proposed operator using an input-operator-output schematic diagram - similar in vein to that used in the previous chapter. It provides a reference of the basic structure of ARROW.

Figure 4.4 summarises the four key stages in the ARROW algorithm in terms of the input, output and algorithmic steps undertaken for each.

	Slice	Dice	Rail	Clip
Input	- Unstructured Points (x,y,z,nx,ny,nz,r,g,b)	- Salient Interest Points	- Element Point Clusters	- Wall Points' Polygon - Surface Element Rails
Algorithmic Stages	- Filter input points w.r.t. their 'verticality' - Localise on wall using Linear-Least-Squares plane of filtered points - Signed-Distance-Field binary pointset split	- Construct KD tree: fast point location queries - Find disjoint connected components in KD tree via AABB optimisations - Remove insignificant connected-components	- Extract extremal frame boundary vector shape - Extract interior sash split-logic axes - Construct 3D model using a set of generalised cylinders	- Divide polygonised wall representation: quadrilateral chunks - Clip quadded chunks against each surface element frame shapes
Output	- Wall Points - Salient Interest Points	- Element Point Clusters: Connected Components	- Railed Surface Elements	- Cutout Wall Mesh

Figure 4.4: *key stages in automatic facade reconstruction from unstructured ground laser-scans in tabular form - each column summarises the details of a particular stage and the order of reading is top-to-bottom, left-to-right.*

Note the manner in which the outcome of each of ARROW's algorithmic stages feeds each subsequent stage. This is in many ways reminiscent of the behaviour of the MAMMAL algorithm defined in the previous chapter.

Pseudo-Code

High level pseudo-code is provided in figure 4.5 in order to summarise the logic of the algorithmic steps that transform a set of unstructured points into a sparse 3D facade model. Note that relative to the MAMMAL algorithm, this pseudo-code is much simpler to understand, due largely to the fact that the scope of the problem is constrained to reconstructing a single facade. Essentially unlike MAMMAL (which is responsible for segmenting, filtering and modelling a city-scale dataset composed of multiple buildings), the expectation here is that ARROW is executed on a single facade scan.

```
1  filter points based on verticality
2  localise on wall using filtered points
3  binary split pointset : wall points vs interest points
4  extract connected-components from interest points using KD-tree
5  for each segmented surface-element
6     generate railed mesh
7  generate quad-dominant cut-out wall mesh
8  return facade : { wall, elements }
```

Figure 4.5: *High-level pseudo-code of the facade reconstructor*

In figure 4.5 type statements are omitted for clarity. The pseudo code clarifies the algorithmic steps that will transform the input point-cloud into a facade-model. Lines 1-3 correspond to the *slicing* step, line 4 to the *dicing* step, lines 5-6 to the *railing* step and line 7 to the *clipping* step.

Figure 4.6 depicts an example of the input data to the ARROW algorithm - an unstructured facade point-cloud a subset of the Roslyn Mews dataset.

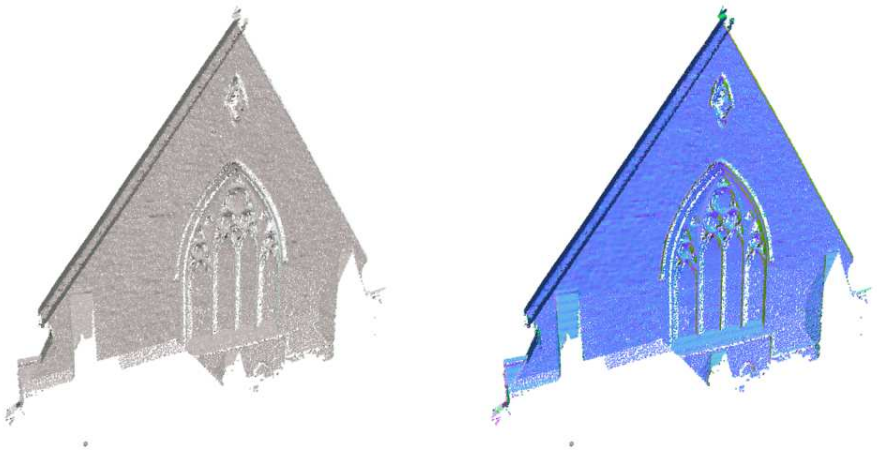

Figure 4.6: *input facade points (left) and points rendered with normal shader (right) for a subset of the unstructured Rosslyn Mews ground-scan dataset*

Essentially this is an algorithm to turn facade-points into facade-models. It is not a general purpose reconstruction method. It serves a niche within the architectural and geometric communities for which accurate automatic detection of window and door apertures and corresponding model generation is highly advantageous. However this also means that if the input does not correspond to a facade scan then the output will fail to characterise the input effectively. The facade-scan in figure 4.6 indicates the class of point-cloud that should be supplied as input. Yes there is still sensing noise present (especially around the edges of the set), and partial/missing data (the occluded region beneath the central window), however the points corresponding to vegetation or street furniture are negligible. In this sense it is a relatively *clean* representation of the facade. Figure 4.6 is a visual guide as to the coverage and density of scans that should be supplied as input.

Having outlined the ARROW algorithm, the next sections detail the internal logistics of the four key stages - the slice, the dice, the rail and the clip. The aim is to provide a comprehensive account of how each stage in ARROW supports the goal or recovering clean, compact, accurate facade models from ground-based 3D laser scans. For each of these key stages the discussions seek to explain what the stage entails, why it is necessary and how it is actually works. In addition the input and output, key-factors, context, relevance and limitations of each processing stage are detailed.

4.3.2 Slice

The initial slicing stage of ARROW is a binary point-set division function that aims to divide the unstructured input facade points into two subsets - with one subset containing points corresponding to walls and the other containing points corresponding to salient features of the facade.

The slicing stage operates by localising on the position of the wall by computing the linear-least-squares wall estimate using a subset of the input points (which are filtered by their verticality). Then it considers the signed distance between each point and the wall estimate in order to isolate points whose distance to the wall is greater than the user-supplied iso-distance.

The context of the slicing stage is in the segmentation of unstructured point-sets. In particular it is a low-level operator for facade division. In a sense, it is analogous to the difference of elevation models stage employed in the aerial change-detection and reconstruction operators, because it results in a binary division such that the points corresponding to salient features are separated from the point corresponding to background/terrain/wall.

The underlying insight is that architectural surface-elements (windows and doors) rarely lie completely flush with a wall. Typically there is depth variance around the geometry of the frames, sashes and sills that results in an inset or outset. As such one can exploit this physical characteristic to frame the problem of salient point detection (given unstructured points representing a planar facade). At a high level the key is that where there are points deviating from the wall, there are points of potential interest.

As such there are two sub-steps employed to slice the input facade scan:

1. Localise on the wall - by computing the LLS error minimising plane using a subset of the inputs points - which are filtered by the verticality of their surface-normal (I') such that $\forall e \in I' : \Delta e_y < \theta$.

2. Label each point as salient or wall - by calculating the distance of each point in the input ($e \in I$) to the wall position estimate (W): $\|W - e\|$.

One may query: why is this relevant or necessary? The relevance of the slicing stage lies in the fact that it is responsible for determining two classes of subset - one that feeds the later *railing* stage and the other that feeds the *clipping* stage. The vital aspect is that this process breaks down an initially complex task into two distinct and (as a result) more manageable sub-tasks. The slice essentially enables ARROW to reform the problem of facade reconstruction as that of wall modelling and feature modelling.

Key-Note: Regarding the use of a signed-distance-field split as opposed to the simpler plane-side split. During the course of the methodology a plane based wall representation is used. However it is important to be aware that this is an implementation detail. In principle the underlying theoretical representation of a wall should be based on a 2D or 3D signed distance field such that the sign of the distance for points on one side of the representation is negative and on the other side positive. It happens to be that in practice a large proportion of walls can be defined with a single linear edge. However in order for the slicing stage to be flexible enough to handle irregular walls, it must based on the notion of querying a general 2D or 3D scalar field representation. This is discussed in greater depth

later in this section.

Formally the input to the slicing stage is an unstructured point-set of a facade I. The intermediary verticality filter subset is I'. The output of the slice operation (O) is two unstructured point-sets - each a subset of the input such that each point in the input is returned as either a wall point or a salient point. Essentially: $\forall p_i \in P : p_i \notin Q$ and vice-versa $\forall q_i \in Q : q_i \notin P$ where: P and Q correspond to the wall subset and salient subset (respectively) and $P \cup Q \equiv I$. The convention used in this thesis is to map O_0 to the wall subset and O_1 to the salient feature subset.

A number of key factors play a large role in controlling the response of the slice. They are briefly discussed in the following outline:

- **Iso-Distance**: which is the threshold used to determine each point's deviation from the wall. If the value is too small it will result in noisy wall points appearing in the salient subset. However if it is too large then window and doors that are nearly flush with the facade's wall are likely to be missed. As a guiding rule-of-thumb it is recommended to start with an iso-distance that is roughly equivalent to $\pm$ 3-12cm - this is based on the physical dimensions of surface-elements, and the assumption of a laser-scan with depth-error of less than $\pm$2cm.

- **Wall representation's support for curvature**: which determines the slicing stage's robustness to different facade wall types. As a result of the verticality constraint there are a number of representational types suited to this. For example a wall could be defined as an on-plan line - which would correspond to a vertical plane. From this a wall could also be defined as an on-plan polyline path - which would correspond to a sequence of vertical planes. Alternatively an implicit curve representation could be defined using an arc or bezier curve. The most generic (flexible) representation would then logically be a general purpose path (composed of polylines and curves). In the case line/plane representations are used, then the wall-descriptor will be composed of piecewise-linear approximations in the case curved and irregular facades - which can degrade the geometric performance of the slice. The way to combat this is to employ the generalised path representation. However the computational expense associated with the slicing operation increases dramatically since the distance to wall function (which is invoked for every point in the input) requires evaluation of a more involved iso-field representation - relative to the simpler (and more efficient) point-to-line/point-to-plane distance measure that is employed by the line/plane sub-case. Figure 4.7 seeks to clarify the notion of the different underlying representations for a wall. The critical point is that each wall representation type is subject to a trade-off between the efficiency with which it can be queried and manipulated algorithmically and its flexibility as a geometric representation.

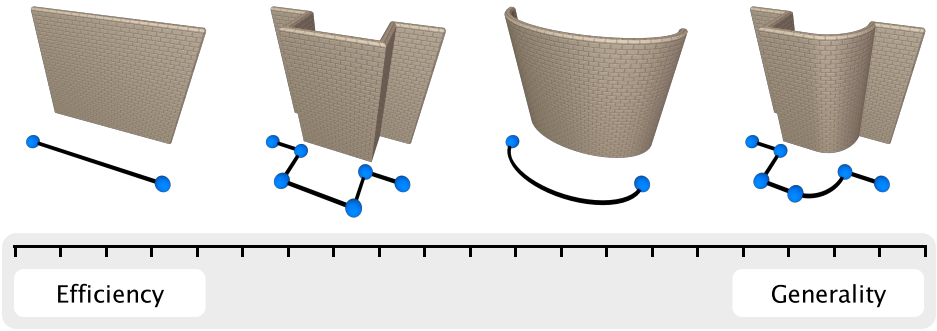

Figure 4.7: *the options for wall representation depicted in 3D (top) and in planimetric 2D (base): illustrating (from left to right): a line representation, a polyline representation, an arc representation and a general path representation - blue points denote the keypoints used to define each wall.*

$$\|(w_0 : w_1) - p_i\| \qquad \begin{matrix} min(\\ \|(w_0 : w_1) - p_i\| \\ \|(w_1 : w_2) - p_i\| \\ ... \\ \|(w_4 : w_5) - p_i\| \\ ) \end{matrix} \qquad \|(w_{distance}(p_i)\| \qquad \begin{matrix} min(\\ \|(w_0 : w_1) - p_i\| \\ \|(w_1 : w_2) - p_i\| \\ \|(w_2 : w_3) - p_i\| \\ \|(w_{3_{distance}}(p_i)\| \\ \|(w_4 : w_5) - p_i\| \\ ) \end{matrix}$$

Figure 4.8: *(from left to right) examples of the corresponding distance calculations required for each type of 2D wall descriptor in figure 4.7 - where* $(w_n : w_{n+1}) - p_i$ *measures the distance between a 2D line and point - and* $w_{n_{distance}}(p_i)$ *determines the distance between an arc and a point.*

It is clear in considering figures 4.7 and 4.8 that as the generality of wall-descriptor increases so to does the expense of its distance measure.

These factors play a large role in controlling the effectiveness of the division that results from the slicing stage. For example a poorly selected iso-distance value will result in a mis-assignment of points.

Beyond these factors there are a number of limitations to the slicing operator which are important to note. They are discussed following:

- The expectation of a reasonably well sampled input facade scan since poorly sampled scans (possessing high point spacing) will degrade the geometric performance of wall-localisation and hence the result of the slice. For example difficulties arise with sparse scans captured from large distances. The underlying limitation is that ARROW requires a dense input point-cloud with high fidelity relative to the physical facade. Execution on sparse scans tends to perform badly.
- The requirement for *verticality* in order to efficiently filter the input points. The limitation being that ARROW could be more effective using an alternative (3D) wall-localisation routine (instead of the planimetric 2D approach) - which would be especially useful for non-euclidean architectural masses. The core limitation is that the constraint enhances the robustness of the wall-localisation phase at the expense of the scope of the types of walls that can be handled. Non-vertical (sloped or angled) walls require an alternative means of filtering (i.e. using a scale space operator such as the DoN[52]).

- The expectation that the majority of the input points correspond to an architectural facade. The slicing operation will not remove clutter in and of itself. It simply divides the point-set into two sets. As such if there are anomalous (non-architectural) components represented in the input they will be preserved in the wall subset and salient subset.

Despite the constraints on the quality of the input scan points thats results from the direct geometric nature of the slice, the advantages relative to stochastic or pattern-recognition based methods are determinism, efficiency and independence from training strategies or datasets.

The following pseudo-code provides an algorithmic perspective of the inner-workings of the initial binary division operation that breaks the problem of facade reconstruction down into two simpler distinct sub-tasks.

$filtered \leftarrow filter\_by\_verticality(input)$
$wall \leftarrow estimate\_wall\_position(filtered)$
$wall\_pts \leftarrow \{\}$
$salient\_pts \leftarrow \{\}$
for each $point \in input$ **do**
    $d \leftarrow signed\_distance\_between(point, wall)$
    if $abs(d) < iso\_distance$ **then**
        $add(point, wall\_pts)$
    else
        $add(point, salient\_pts)$
    end if
end for
return $\{wall\_pts, salient\_pts\}$

Or: for each point in the input determine the magnitude of its distance to a linear-least squares wall representation (computed from the subset of input points whose normal indicates verticality) and add points with magnitude $\leq$ iso-distance to the wall subset and $>$ iso-distance to the salient subset.

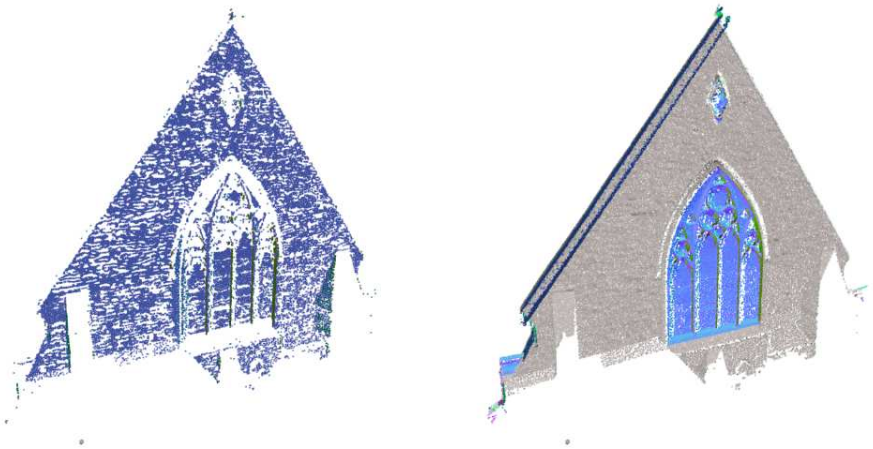

Figure 4.9: *filtered wall points (left) detected using the verticality predicate and the binary division (right) resulting from the signed distance field split - illustrating the outcome of the slicing stage of ARROW: a wall-point subset (in gray) and a salient-point subset (rendered with normal-shader)*

At its heart, this slicing stage is a simple, fast, data-driven, geometric, non-stochastic equivalent of the Difference of Elevation Models' slice operation refactored for unstructured facades point-sets. Figure 4.9 depicts the result of its application to the example input from the Roslyn Mews dataset.

In figure 4.9 note that the verticality filtered subset (left) which is used as intermediary data in order to localise on the wall representation - contains points belonging to both the wall and the salient features. Yet this does not break the subsequent SDF-split's result (right) which clearly separates the salient parts of the scan (the surface elements) based solely on the deviation from the estimated wall with an iso-distance threshold of $\pm$5cm.

The following enumerates (for reference) the key *slicing* concepts discussed.

Wall Localisation

- Local Point Level Operation - yields efficiency relative to region or neighbourhood level operators (such as the Difference of Normals [52]).
- Verticality Filtering - does not rely on consistently oriented normals - enables use of the linear least square plane by omitting outliers which could perturb the position of the wall - a product of the observation that the large majority of architectural walls are vertical.

Signed Distance Field Segmentation

- Automatic Point Assignment: decides which points should belong to the wall subset and which to the salient subset.
- Efficient ways to implement the point-to-wall distance-function for the wall descriptors supporting curved and irregular arrangements.

Once the wall and salient subsets have been assigned, the next step in the ARROW algorithm is to segment the salient subset in order to isolate individual surface elements. The *dicing* operation is discussed next.

4.3.3 Dice

The next stage of ARROW is the detection and segmentation of individual windows and doors point-clusters from the salient subset identified by the previous slice operation. The aim is to isolate groups of points that correspond to surface-elements such that each can be processed independently.

In essence the dicing step is similar in vien to the connected-component extraction used in the aerial operator's segmentation. The key difference being that here we are extracting these components from an unstructured scan. Nonetheless, the core concept is much the same. The dicing step first determines local (point-level) neighbourhood connectivity. Then breadth-first traversal of the graph constructed from the salient subset's neighbourhoods.

The critical problem with this stage is that point-location and spatial queries are proportionally a lot more expensive with unstructured scans relative to performing equivalent operations on the structured aerial range images.

To address this the dicing stage exploits a KD-tree (in order to *chunk* the salient subset), AABB spatial optimisations (in order to quickly dismiss groups of points that are not connected), and a bottom-up graph-labelling routine (in order efficiently traverse and resolve components).

Remember the context is of this sub-problem is object segmentation from unstructured point-clouds, and in particular automatically identifying and isolating the structural surface components of a building facade.

In terms of the relevance of the dicing: it is the key to breaking down the salient parts of the input facade-scan into manageable contiguous clusters which can be fed directly into an automatic window reconstruction routine. Interestingly (as a result) this stage could also be considered as analogous/equivalent to the maximal-area roof-shape segmentation employed during airborne reconstruction - in the sense that it breaks each facade's un-ordered point descriptor into sub-objects (individual components). The only semantic difference (which makes the comparison with the DoEM connected component extraction more adequate) is that the MARS algorithm cannot rely upon the disjointness of roof-shapes in order to find a division - whereas this dicing stage can (exploit the disjointness of surface-elements). This is a product of the *axis-aligned-disjointness* principal/heuristic that frequently pervades groups of windows and doors upon architectural surfaces.

The high-level algorithmic steps that implement the dice operation are:

1. **Construct KD-tree**
2. **Label neighbouring chunks in KD-tree**
3. **Extract connected components using neighbour chunks**
4. **Filter anomalous and insignificant components**

Formally the input to the dicing stage is the salient feature points derived from the signed distance field split (a subset of the input) - denoted P. The output of dicing is a set of point-clusters Q, each a subset of the salient set (such that $\forall qi \in Q : qs \subset P$ and $\forall qi_s \in qi : qi_s \notin qj, \forall j \leftarrow [0:n] : j \neq i$.

Essentially the input salient feature points are broken down into a set of disjoint connected components within which each point in the input is assigned to a single connected component. Each connected component maps to a cluster of points representing a window or door (surface-element) and the connected-components are each non-overlapping regions in the input.

The key factors that control the efficacy of the dicing stage are:

- **Point-Location**: or rather the computational performance properties of the point-location strategy used - most vitally whether it results in constant-time or variable-time (data-dependant) point queries and whether it produces exact or approximate responses to queries.
- **Spatial-Optimisation**: which is used to reduce the number of queries required in order to perform operations such as finding a point's nearest neighbour(s), by using a structure (oct/K-d tree - an additional layer of abstraction in the point-cloud's representation) that recursively divides the input spatially in order to access points or regions more efficiently as a tree. There are

also non-recursive spatial optimisations which include hashing and bucketing strategies - they are generally faster to construct and interrogate, but less spatially adaptive to the data.

- **Point-Set Intersection Test**: which is the connectivity constraint (or rather the predicate) used to determine whether two clusters of points are connected - i.e. the function used to answer the question: are these two clusters of points part of the same surface element? (Are point-sets A and B, connected, contiguous, colliding?)
- **Filtering and Post-Processing**: which determines whether there is any scope for refactoring the result post-execution to address anomalies (over and under segmentation) in the extracted clusters.
- **Quantifying Detection Recall and Precision**: in particular this relates to the manner in which ones measures how effective each segmentation result is in terms of accurately representing the structural decomposition of each facade - which calls for ground-truth data.

Ultimately the path taken in this research is towards purely data-driven window and door detection from unstructured point-sets. As such the benefits of the dicing strategy proposed include deterministic (non-stochastic) region results and repeatable computational performance. Further the localisation of window and door point-clusters executes more efficiently than the current state of the art [77][78][21]. The critical point is that the KD-tree and AABB spatial optimisations yield segmentation results analogous to using the prevalent facade segmentation methods but in a fraction of the time.

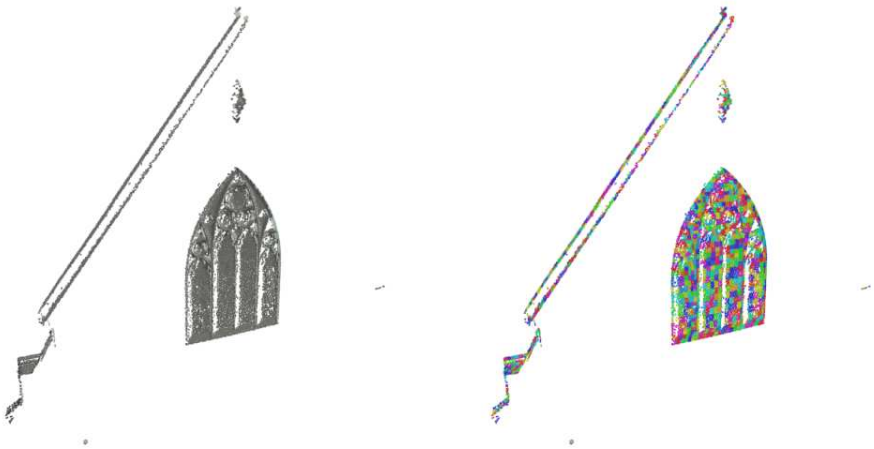

Figure 4.10: *the KD-tree chunks (right) constructed from the salient subset of points (left), for the example facade used throughout this chapter*

The underlying insight is that KD-trees exhibit lower recursive growth (n^2, relative to oct-trees: n^8), yet yield effective chunking results due to the largely planar nature of facade-scans. This means far fewer *chunks* are generated to dice-up the salient points - which greatly enhances efficiency without sacrificing segmentation quality. This also relies on the disjointness of architectural surface-elements discussed earlier. Further the addition of the axis-aligned bounding boxes (AABBs) allows

the routine to quickly dismiss chunks that cannot possibly belong to the same object (without the greater expense of comparing the distance between clusters point for point). This has the effect of reducing the number of distance calculations required.

Figure 4.11 illustrates the outcome of the dice on the example facade-scan.

Figure 4.11: *the connected components extracted from the salient subset rendered in isolation (left) and alongside the wall-subset (right) - rendered using a pseudo-stochastic mapping between component-id and point colour*

Another positive feature resulting from the data-driven nature of the slice is that it is capable of returning irregularly shaped surface-element point-sets. Commonly box heuristics are used [93][8][153][139][63][110][152], which although suitable for the large majority of windows and doors, perform poorly on irregular features (such as the central window in figure 4.11). The fundamental advantage of a data-driven (over a templated or heuristic) dicing method is that it represents the data that is actually present in the salient subset, however varied.

Despite these positive operational properties, the ideology is not perfect and the performance of the dicing stage is subject to the following limitations:

- Expectation of a reasonably high-resolution scan (as in the slice).
- Dependant on the efficacy of preceding slice stage's binary split.
- The data-driven nature means that it does not fill in holes where there is missing or partial data - i.e. a partially scanned window that appears as two distinct clusters of points will yield two distinct components.
- Resolving over and under-clusters components - (in particular over-clustered regions) adds another level in the hierarchy of the facade's segmentation that cannot rely solely on disjointness.

Most of these limitations result from the data-driven nature of the operation. However there are also aspects that require further research in order to be adequately addressed. In particular improvements to the post-processing of the *diced* up element point-clusters would help to deal with under and over clustering. Further the use of additional spatial measures could constrain the segmentation result - and

help in dealing with *broken* point-sets - which are geometrically distinct (disjoint) but semantically belong to the same object. Such occurrences are generally the product of partial data and are manifestations of under-clustering. On the other hand, to limit over clustering, one could pre-filter any wall points that may have slipped through the preceding SDF split into the salient set. This would help mitigate over-clustering by preventing disjoint element-clusters appearing to be contiguous because wall points *join* them together. There are in essence a myriad of operations that could be applied to improve the division resulting from the dicing stage. The key point is that each such operation will either seek (whether heuristically or analytically) to prevent the problem of over or under clustering.

Having provided an overview of the dicing stage: the following covers the internals of the specialised connected-component routine used to efficiently traverse the unstructured data and isolate windows and doors.

Efficiently Clustering Connected Components

The last relevant detail of the dicing stage is how to efficiently traverse the KD-chunks to produce connected-components such as those illustrated in figure 4.11. The core idea is that one treats the problem as a graph traversal task - within which the aim is to isolate the minimum spanning trees of the connectivity graph of the *chunked* components. The efficiency of this strategy is a product of the layer of abstraction introduced between the input salient subset and the connected-component routine. By stitching together groups of related *chunks* as opposed to points the expense of isolating connected-regions in the unstructured input is significantly reduced.

Note: though that this efficiency comes at the slight cost of precision since - in principle more precise connected components could be recovered using an exact point-for-point approach. However in practice the difference in accuracy is typically negligible given the nature of architectural facades, and for this particular application at least (fast, accurate and sparse reconstruction), generally not worth the dramatic increase and execution time.

To revise: the *dicing* stage of the ARROW algorithm automatically divides the set of salient feature points into individual *surface-element* point-clusters, such that each may be analysed and processed independently. The next stage is the *rail* which involves iterating over each surface-element point-cluster and constructing a 3D window or door model to represent the geometric form of the element. This is discussed next.

4.3.4 Rail

The rail stage of the ARROW algorithm transforms each cluster of segmented surface-element points into a sparse 3D model. It is in essence an automatic modelling stage for window and door point-sets. The key aim is to create accurate and compact 3D models from a cluster of points - assumed to belong to an architectural opening, surface feature or adornment.

To address this problem a novel procedural representation of the an architectural surface element is proposed and exploited experimentally in order to produce data-driven 3D models from segmented point-clusters. The *Surface-Element* is simply a formal representation of the structure of an architectural opening that exploits sets of generalised cylinders (3D sweeps) in order to model depth variance around frames, sashes and sills in tandem with 2D procedural split-logics - that provide an abstracted means of defining the division of space that characterises each surface-element. The key benefit of the representation is that it provides a vital layer of abstraction between the specification of the semantic structure of an element and its realisation as a 3D model. This layer of abstraction, coupled with the Surface-Element's generality makes it suitable for forward-chaining (user-centric) and backward-chaining (reconstructive) modelling tasks. The underlying idea is to encode the process of modelling a window or door in a data-driven function - such that a wide array of varying window and doors can be dynamically generated from sparse semantic descriptors.

Theoretically the Surface-Element is a mapping between a set of 2D vectors and a 3D polyhedral mesh. Implementation-wise: it is simply a function that takes 2D shapes as input and constructs and returns a 3D model.

The primary reason it exists is to help combat the problems inherent to template-based and pattern-matching methods of window and door reconstruction - which provide approximate results. This is the key limitation of the pre-existing model based strategies [93][152]. The issue is that irregular windows and doors (for which effective approximates do not exist in the scheme's *knowledge-base*) cannot be correctly reconstructed with model based strategies. One way to combat this is to increase the scope of the scheme's knowledge-base (add more templates or patterns to the library) [95]. However the unavoidable disadvantage of this is greater computational load. Furthermore even with a large knowledge-base (library of common models) to search, ultimately there are still no guarantees of a correct result.

The logical way to combat this is to either faithfully reconstruct the data without reference to a library of common models, or alternatively devise a means of altering items in the library dynamically to better fit the input.

On the other hand (at the other end of the ideological scale) purely data-driven methods of window and door reconstruction tend to result in dense, lower-quality models, that are often not suitable for texturing, rendering or animating. Although for analytic tasks geometric accuracy is the deciding factor, practically the lack of semantic structure limits the use of dense surface-element models for tasks such as realtime interactive display.

Fundamentally data-driven methods are commonly plagued by the lower quality of the resultant models, whilst template-driven methods are most often limited by the scope of the features they can accurately represent. Ultimately though the desire

is for the computational efficiency and accuracy of data-driven methods with the model quality characteristics of a template-driven method. Hence this research devised the Surface Element - a procedural representation designed to support the pursuit of these desires.

The body of this section is structured as follows. It first introduces the Surface-Element as a general purpose window and door representation, with clarifying examples. It then provides a formal definition of the abstraction. It then considers the problem of automatically constructing surface elements descriptors from each cluster of segmented points in the broken down salient subset and proposes a simple algorithm (SLADE - Split-Logic-Axes Detector and Extractor) to address the problem of split-logic resolution.

Architectural Surface Elements: An Overview with Examples

Put simply: a surface-element is an object (element) that resides upon a surface. Hence an architectural surface-element is an object that resides on an architectural surface. In layman's terms here elements refers to windows and doors whilst surface refers to a building facade.

Note: *throughout this chapter, the term Surface-Element is used to refer specifically (and exclusively) to architectural surface-elements.*

The idea is that although individual windows and doors can vary greatly in their appearance and structure, they all belong to the same class of entity. As such there are common features of the class which are useful in characterising the class at a high-level. To put it another way, there are fundamental similarities that pervade windows and doors - due largely to their function as openings. One can exploit these similarities in order to describe and manipulate the notion of a window or door abstractly.

As an example windows and doors typically possess a boundary (or frame) shape. They also possess a character-giving division of space (in the form of sashes and panes). They can also be bordered by sills or ledges (or steps in the case doors). They may optionally possess behavioural attributes such as the nature of the opening transform (translational or rotational).

This initial example break-down sets the tone for the rest of this discussion. Note that structurally each surface-element is a composite of different types of sub-component (frame, sashes, panes, sills...).

Note: *in the following portions of this section a distinction is made between **regular** and **irregular** surface-elements. It is imperative that the reader is aware that this distinction exists solely to aid in the exposition of the representation and operations employed by the railing stage. In principle there is no such dichotomy. The formal definition and the function implemented to realise the abstraction recognise no such distinction. However in practice it is quite useful to be able to treat elements as either regular (easy cases) or irregular (complex cases) - because it enables the resolution of the large majority of elements to be optimised to deal with axis-aligned structural edges. This explanatory distinction is discussed in greater detail shortly.*

Rectilinear/Regular Surface-Elements

Regular surface-elements are composed of axis-aligned vector edges. The essential attribute is that the semantic structure of a regular surface-element can be defined accurately using purely rectilinear components or horizontal and vertical edges. Regular surface-elements are the most common form of architectural opening. Figure 4.12 illustrates regular surface elements.

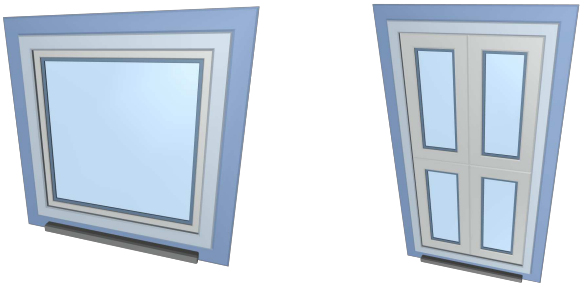

Figure 4.12: *examples of regular, rectilinear procedural surface-element models: depicting (from left to right) a window, a door and a balcony.*

Note that in figure 4.12 the bounding box of each element is equivalent to the element's frame shape and the internal sash arrangement possesses only horizontal or vertical edges. Regular surface-elements are useful because of the simplicity with which they can be defined and the frequency with which they occur in real environments. However they are limited to describing box-structured windows and doors. As such in order to handle the diversity present in architecture, non-rectilinear descriptors are also required.

Non-Rectilinear/Irregular Surface-Elements

Irregular surface-elements are composed of non-rectilinear vector shapes. The essential attribute of an irregular surface element is that the semantic structure cannot be accurately defined using purely rectilinear components. One requires polygonal descriptors to effectively model an irregular surface-element. Irregular surface-elements occur less frequently upon facades however they are common in ecclesiastic architecture. Although the prevalence of irregular surface elements varies from region to region, the fact that they exist mandates their inclusion in this scheme. Figure 4.13 illustrates examples of irregular surface element models to clarify.

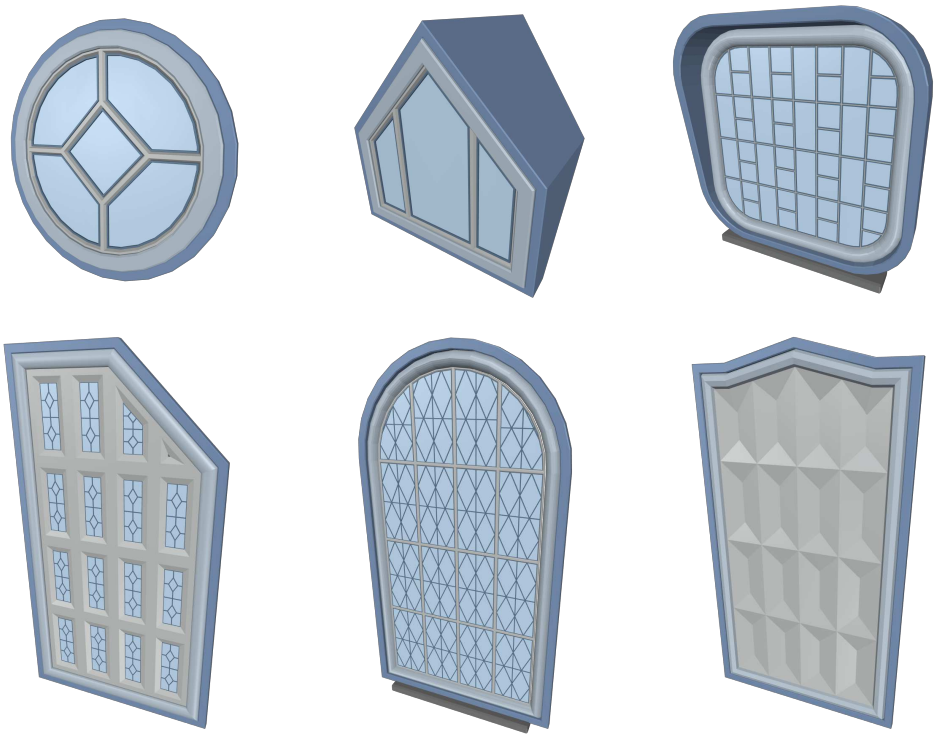

Figure 4.13: *examples of irregular, non-rectilinear procedural surface-element models: depicting (from left to right, top to bottom), a circular window with polygonal sash-division, a braced window with rectilinear sash-division, a round-rectangle braced window with tiled (repeated) sash-division, a bevel-corner door with nested sash and detail divisions, a round arch window with nested sash and detail divisions and an opaque door.*

Note that in figure 4.13 each element possesses either a non-rectilinear boundary, internal division or both. Essentially one could not represent the structure of such surface elements using only vertical and horizontal edges. These irregular surface elements also indicate different types of depth variance.

Vitally though, despite this distinction, the thing that is common to all is that they are characterised by a boundary shape and an interior division (splits). Secondary characteristics include profile shapes and the scale of each component. The key is that in order to define and manipulate each of the examples there are only two types of objects that require support. This representation relies on two basic primitives: shapes and split-logics.

Shapes

In this scheme shapes are 2D vertex-lists that describe polygonal forms.

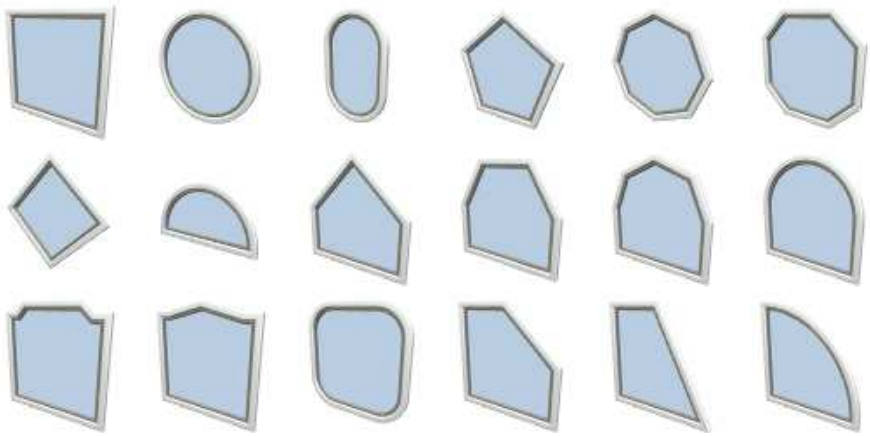

Figure 4.14: *Examples of different shapes supplied as input frames to generate single-sash window models. Each shape (a 2D planar simple polygon) can be used to represent both the object-boundary and the object's set of defining profile vectors. The key consideration in deciding upon a shape representation is whether it is explicit (i.e. a static set of vertices) or dynamic (as in a function generating a set of vertices). Alongside split-logics, shapes are one of the fundamental geometric primitives exploited by the proposed scheme - and by procedurally combining them with split-logics, generate 3D models.*

Split-Logics

In this scheme split-logics are functions that encode a division of 2D space. They can be applied recursively, selectively and/or conditionally.

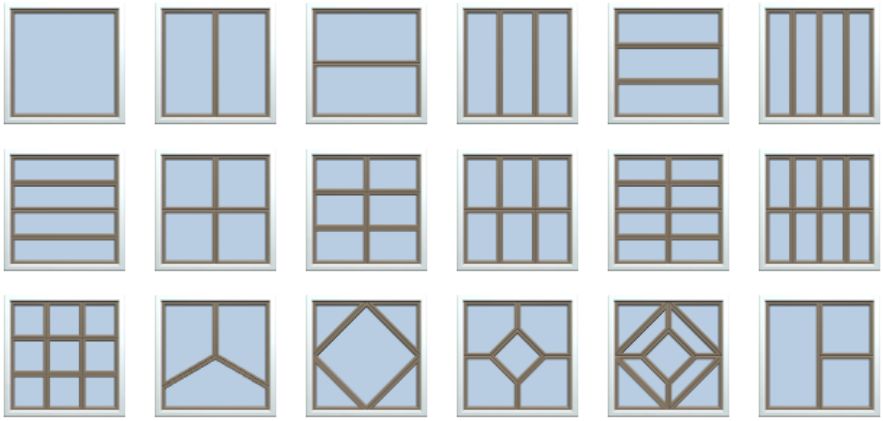

Figure 4.15: *Examples of different 2D split-logics applied by the method in order to generate regular rectilinear and irregular sub-divided sash layouts. Each split-logic defines the manner in which sub-regions should be created from an input parent region (which in these examples is a square of unit length). Each is a function that encodes a characteristic division of space which defines the internal structure of a generated surface-element.*

Formal Definition of a Surface-Element

$$E \leftarrow \{(F_S \wedge F_P) \cup ((S_P \wedge S_L()) : \forall c[i] \in S_L(F_S))\}$$

where:

- E : denotes the resultant surface-element a set of generalised cylinders
- F_S : is the polygonal frame shape - a cyclic 2D vertex-list (the outer hull of the generated surface-element: its defining boundary)
- F_P : is the frame's profile shape - a cyclic 2D vertex-list (the depth)
- S_P : is the shared internal sash profile - a cyclic 2D vertex-list
- S_L : is the split-logic function that defines a division of 2D space - normalised to the lie in the range [0:1] along the principle axes (X, Y)
- $c[0-n]$: is the set of sub-regions derived from applying the split-logic function (S_L) to the input frame shape (F_S) - the result of sub-division
- The $'\wedge'$ operators is used abstractly to represent a multiplicative operation in the geometric sense. It computes the pseudo product of two input shapes in R2 and yields a polyhedral mesh in R3. Here it effectively symbolises the 'sweep' (generalised-cylinder) function.

for example:

$$E \leftarrow \{(fs \wedge fp) \cup (sp \wedge sl(fs))\}$$

where:

- $fs \leftarrow \{[-0.5, 0], [-0.5, 1], [0.5, 1], [0.5, 0]\} \times [width, height]$
- $fp \leftarrow (\{[-0.45, -0.5], [-0.5, -0.45], [-0.5, 0.45], [-0.45, 0.5],$ $[0.45, 0.5], [0.5, 0.45], [0.5, -0.45], [0.45, -0.45]\} - [0.5, 0]) \times \vec{fp_{scale}}$
- $sp \leftarrow (\{[-0.45, -0.5], [-0.5, -0.45], [-0.5, 0.45], [-0.45, 0.5],$ $[0.45, 0.5], [0.5, 0.45], [0.5, -0.45], [0.45, -0.45]\} + [0.5, 0]) \times \vec{sp_{scale}}$
- $sl \leftarrow \{\{[-0.5, 0], [-0.5, 1], [0, 1], [0, 0]\}, \{[0, 0], [0, 1], [0.5, 1], [0.5, 0]\}\}$

yields the 3D model depicted in figure 4.16:

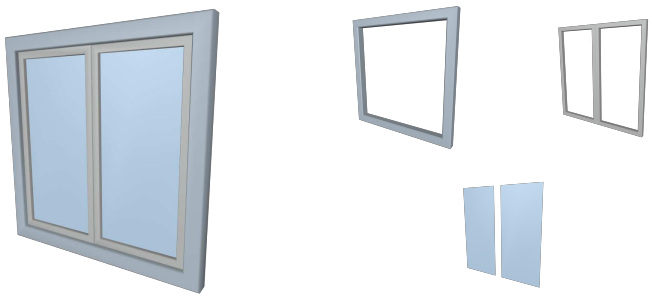

Figure 4.16: *window model generated from the simple example surface-element (left) with its constituent components: frame, sashes, panes (to the right)*

From this additional components can be instantiated by altering the simple definition to include supplementary shape and split-logic operations. Three such extensions are outlined. First the recursive application of detail split logics, secondly a frame-brace and thirdly north-west-east-south (NWES) sill rails. They aim to enhance the representation's ability to characterise the diversity present in real-life surface-elements by mapping the most common features to manipulatable components. They simply extend the representation with data-driven 3D sweeps.

The detail split-function is applied to each sash sub-division pane in order to create nested divisions of space. This component enables seemingly complex 2D patterns to be constructed through the combination of simple split-logic functions. Just like the sash component the detail component is defined by a profile shape (DP) and a split function ($DL()$). Note that whilst the structure of each surface-element's planimetric division is probably best represented using a tree structure, in-practice only 2 or 3 levels or recursion are typically required to describe the large majority of surface elements. This extension adds a single recursive layer to the sash division layer. This limitation seeks to simplify the definition and enhance its clarity: and though it is trivial to add additional layers, ultimately one must balance the representation's scope for characterisation relative to flexibility of manipulating it (both manually and procedurally). Nonetheless the underlying concept is easy to grasp: essentially rather than introducing increasingly complex split logics (in order to model irregular features), one can simply combine simpler (somewhat axiomatic) split logics dynamically. Again the key idea is to encode self-similar patterns using simple split-logics. As such if the set of sub-divided sash shapes (SS) is given by the expression:

$$((S_P \wedge S_L()) : \forall c[i] \in S_L(F_S))$$

then the corresponding detail mesh is constructed as a product of:

$$\forall\ s_i \in SS : E \leftarrow E \cup \{(D_P \wedge D_L()) : \forall c[j] \in D_L(s_i)\}$$

which yields nested sash divisions such as those depicted in figure 4.17.

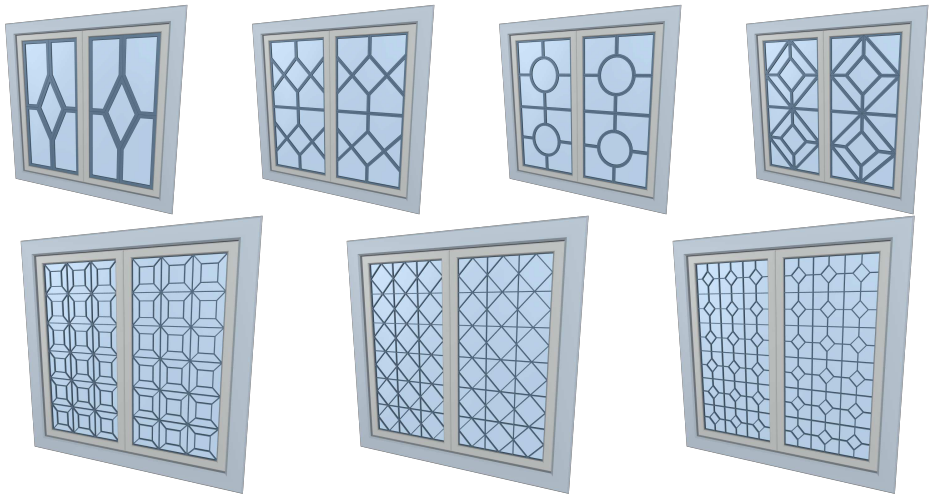

Figure 4.17: *examples of tiled detail split-logics applied to the example surface elements two component sash split-logic to represent various window's panes*

Note: in these examples that the nested (recursive) split function employed is not the only variable - but that we can also apply instancing logic (repetition) and non uniform scaling to alter the characteristics of each window.

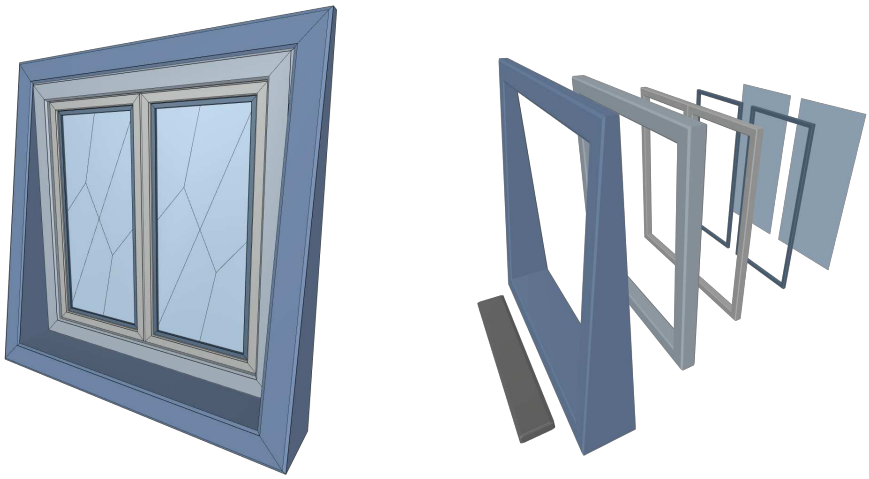

Figure 4.18: *example surface element with brace, detail and sill components: illustrating (left) the merged model and (right) the individual components.*

Figures 4.18 and 4.19 demonstrate additional aspects of the abstraction.

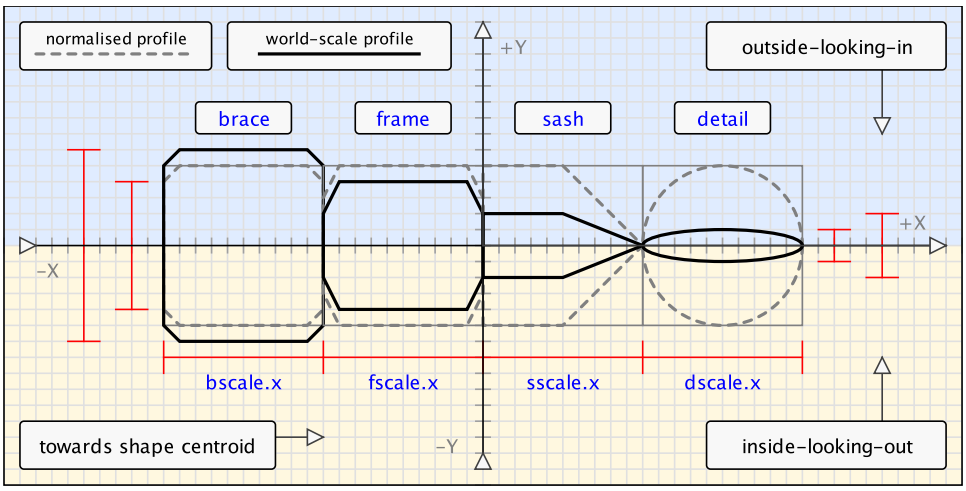

Figure 4.19: *An explanatory two-dimensional slice representation of a surface-element's defining profiles vectors - depicting the structural composition of a generated window model's geometric form in terms of a set of 2D planar profile-shapes that are 'swept' along input 'rail' shapes to create depth. This figure also clarifies the axis conventions utilised - with the positive X direction mapping to moving 'inwards' towards the centroid of the input shape.*

Note: that the conventions in figure 4.19 are by no means the only valid means of specifying the relative position of sub-components. Indeed one could just as easily invert the X or Y axis order to yield alternative specifications. The critical point is that irrespective of the conventions employed - one should be consistent in the manner of offsetting components within both forward-chaining (user-centric interactive surface element creation) and backward chaining (reconstructive) contexts.

Having explained what a surface element is - and the key features of the abstraction - the next section deals with how to extract simple surface-element descriptors from the segmented window and door point-clusters.

Automatically Recovering Surface-Elements

This section discusses the means of turning each cluster of window and door points into a structured (component based) 3D model. It proposes a 3-stage operator dubbed *SLADE* (the Split-Logic-Axis Detector & Extractor).

To address this problem - the SLADE operator first extracts extremal shape boundaries for each point-cluster (to determine frame shapes) and then resolves each under a planimetric arrangement (to derive interior split-logic axes). Split-logic functions are then constructed and applied to the frame boundaries to produce sash components which (along with the frame) are modelled as swept profiles (3D-sweeps, generalised cylinders).

Formally the input to the *rail* is a cluster of 3D points of a segmented surface-element (SE). The output is a sparse 3D model (represented as a polygon-mesh) and a 2D planimetric vector descriptor of the element - both representations of the semantic structure of the input point-cluster.

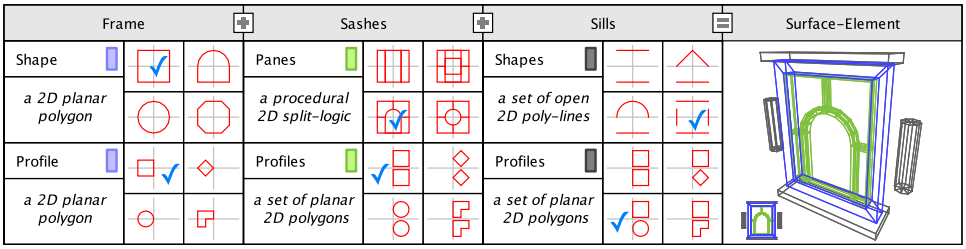

Figure 4.20: *schematic overview of the basic operation of SLADE - illustrating the three surface-element component types sought - alongside the characteristics of the 2D descriptors that are employed for each component type.*

Although figure 4.20 illustrates some of the potential 2D geometries that could be returned - in practice SLADE operates in a data-driven manner. Essentially - whilst one could construct surface-elements in a templated manner - the most accurate results are the product of mining the descriptive vector shapes directly from the point-sets. The following outlines how.

- **Detecting Primary Frame** - by extracting a planimetric 2D hull of a cluster of points via a normalising (flattening) projection - which is then reverted to return to the hull to the 3D coordinate system.
- **Detecting Interior Panes** - by extracting interior edges, extending them outwards to intersect with the frame-shape and clipping the frame shape about the interior edges to yield sub-divided panes.
- **Detecting Auxiliary Components** - (such as window-sills) as edges.

Once surface-element descriptors have been extracted - ARROW generates 3D swept profiles and feeds the 2D vector descriptors to the dicing stage to produce aperture (cut-out) facade wall models - discussed next.

4.3.5 Clip

The last stage of the ARROW algorithm is the *clip*. At its heart, it is a method of creating sparse *cut-out* wall meshes with holes in position for each surface-element. The *clipping* stage exists in order to transform the subset of wall points (from the initial slicing stage) into a quad-dominant 3D model that is light-weight enough to be suitable for interactive simulation and that accurately reflects the apertures upon the facade in order to facilitate analytic tasks such as solar-gain calculation.

This stage is essentially a sparse wall meshing operator. It first extracts a vector boundary shape from the wall-point subset. It then sub-divides the polygonal wall representation into quadrilateral pieces. Then (for each quad-piece) it *clips-away* window and door holes using the vector boundaries (the frame shape) of each overlapping reconstructed surface-element.

Formally the input to the rail is the subset of wall-points (produced by the SDF-split) and the frame shapes (extremal planimetric 2D boundaries) of each surface element. The output of the clip is a 3D quad-dominant wall-mesh with holes in place for each surface element's aperture.

The context of this sub-problem is in architectural surface-reconstruction - and it represents a special case of a point-set meshing routine that exploits the primarily planar nature of building facades to support the conventional (industrial) desire for quadrilateral meshes over tetrahedral meshes.

The clipping stage is responsible for completing each facade's geometric model. It is relevant because it enables the resulting geometry to be used in interactive first-person architectural simulations wherein it is necessary to be able to *see-through* (into and out of) surface-elements. Vitally it is a data-driven strategy that seeks to produce sparse (higher-quality) wall reconstruction results than the prevalent RANSAC based strategies.

Although this stage is incredibly simple the novelty lies in its ability to efficiently produce *cut-out* wall meshes - which are not generally supported by the prevalent facade reconstruction or approximation operators. Furthermore few forward (user-centric) systems automatically generate cut-out facades meshes. Generally existing sparse algorithms tend to use a single quad, or a triangulated polygon to represent each facade wall, and simply append window and door models on top without altering the wall mesh's topology. The key problem with such approaches is that they limit the utility of the model that results since there is no trivial means to create transparent features in place for simulating views through windows. This limits the level of realism that can be attained without additional user intervention. Manually editing each wall to introduce apertures is a laborious task that can be better handled automatically. Another benefit of this clipping strategy is that because the geometry is derived from laser-scans (as opposed to photographs), the ability to accurately represent the input makes the wall meshes suitable for physical simulation and building energy analysis.

At a high level the clipping stage performs the following operations:

- Polygonises the input wall-point subset (boundary extraction).
- Sub-divides the wall polygon into axis-aligned quadrilaterals.
- For each sub-divided quadrilateral - clips it relative to each (if there are any) overlapping surface-elements' frame shape(s).

Based on these operations the response (computational and geometric performance) of the slice is largely controlled by the following key factors:

- **Size of quadrilateral chunks**: used to sub-divide the polygonised wall. If they are too large, then a lot more clipping is subsequently required. However if each piece is too small then the size of the output wall mesh (in terms of the number of vertices and faces) will be bloated.
- **Unwrapping of non-linear facades**: which involves projecting the curved and/or piecewise linear wall descriptor and points onto an arbitrary (but consistent) 2D plane so as to enable efficient clipping relative to the 2D planimetric surface-element vector boundaries.
- **Polygon clipping algorithm**: used to handle chunk splitting.
- **Clipping constraints**: that are applied for unwrapped non-linear facade descriptors in order to preserve key-points such as corners.

Ultimately the clipping stage re-frames a common architectural surface-reconstruction problem (that of wall-modelling) as the product of bread-and-butter computer graphics techniques. This is the simplest stage of the ground-reconstruction process. It is intuitive to grasp and incredibly easy to implement. Indeed there is very little that can go wrong and the only potential breaking-point is boundary extraction. However as long as the polygonisation routine used can handle concave shapes, this is generally not a problem. The insight is that by sub-dividing the wall polygon before clipping one can (to a large extent) control the valence of vertices in the result.

So bear in mind that although the quadrilateral sub-division could be considered optional (given that one could also clip the polygonised boundary directly) it actually has a key topological benefit - since it allows the algorithm to control the meshing of parts of walls without surface elements in a manner that is analogous to a human CAD technician.

Additionally there are a number of different ways of implementing an extended version (with an un-wrapper) for non-linear facades. The key is that the each such method reduces the 3D problem down into a more efficiently addressed 2D problem. In essence (irrespective of the type of wall-representation used, single-edge, poly-line, curve or general-path), the underlying idea is to reduce the dimension of the problem to improve efficiency.

Despite the positive aspects of the clipping stage, there are also some down-sides. In terms of its limitations, the biggest issue it that the clip does not automatically decide upon a good quad-piece size. Rather this is left up to an end-user to supply as an input argument. There are two possible conventions for specifying the quad-piece size. One can specify them explicitly (in the form of a width and height - for which the default is 20cm x 20cm), or implicitly (in terms of a maximum number of quad-cells along the x and y axis - which defaults to 24x24). The explicit method allows for strict control of the size (dimensions) of each quad. The implicit method enables one to control the maximum number of quads that will be generated per-facade.

As such one major enhancement would be to automatically determine an effective quad-piece size - possibly by minimising the number of polygonal-faces in the result or considering the quality of the output mesh.

Figure 4.21 illustrates the outcome of of the clipping stage for the example facade used throughout this chapter. The quad-dominant wall-model (right) is produced from the wall-point subset (on the left).

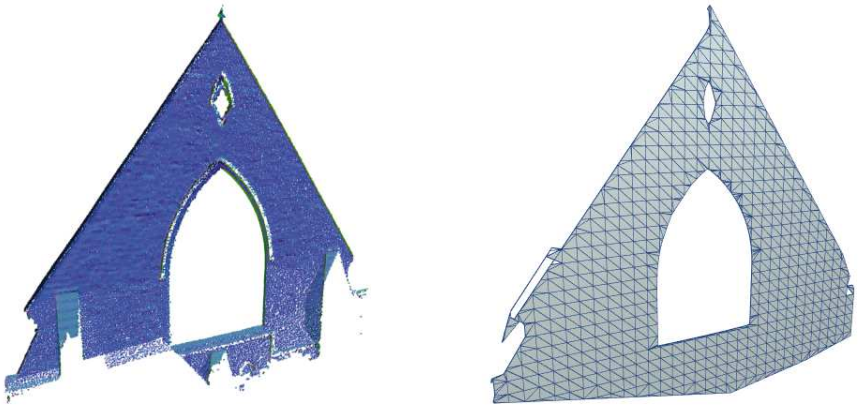

Figure 4.21: *the clipping stage of the ARROW algorithm turns the subset of wall-points (left) into a quad-dominant wall model (right) - the example shown was generated with a quad-piece size derived from a 32x32 grid*

Figure 4.22 summarises the logic used to produce the result in figure 4.21.

Figure 4.22: *logic of the clipping stage summarised: wall boundary (left) minus surface-element boundaries (middle) equals aperture wall model (right)*

Now that you understand what the clip is, why it exists and roughly how it works, the next blocks delve into how to implement it efficiently.

***

Based on the distinction between regular and irregular surface-elements outlined in the previous rail stage, the structural arrangement of a facade can be categorised as either being composed of regular, irregular or a mixture of surface-element types. This categorisation is useful because it enables the types of shape used during the clip to be restricted to enhance efficiency.

Facades Composed of Regular Surface-Elements

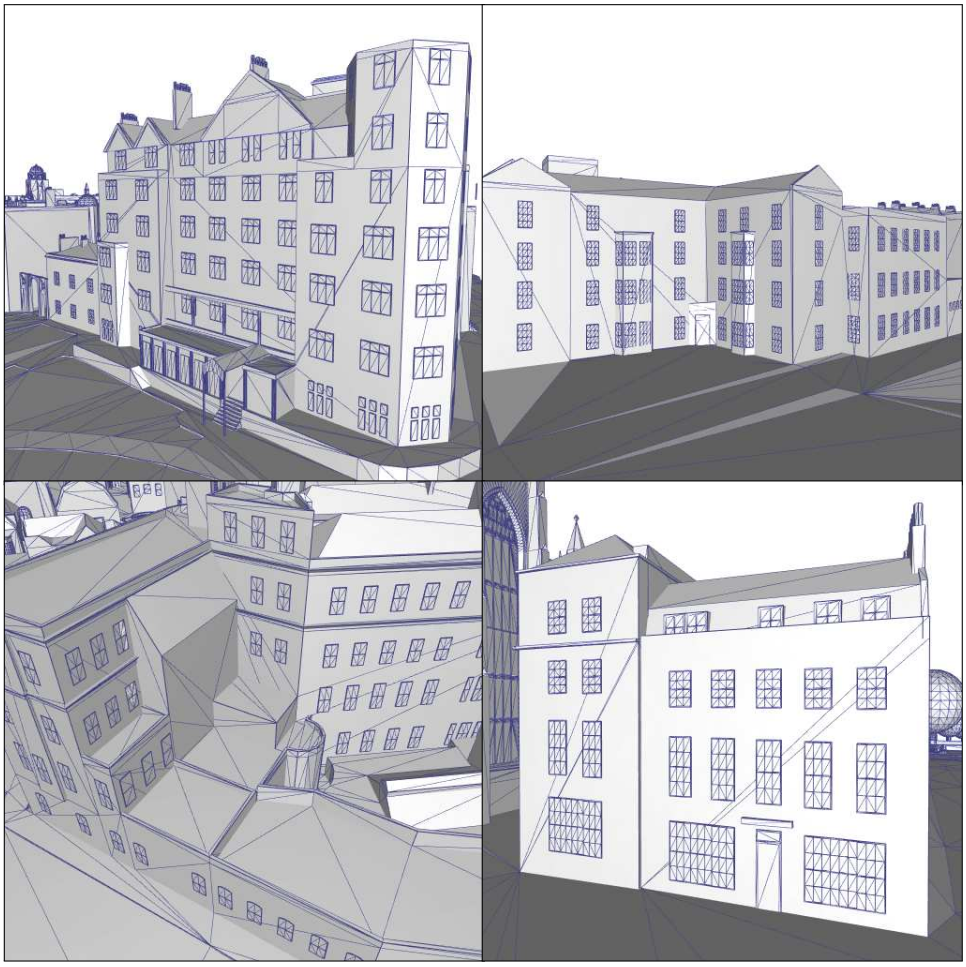

Figure 4.23: *Examples of manually constructed facades composed entirely of regular rectilinear surface-elements from four different buildings in the City of Bath CAD model - rendered with lines to display mesh topology.*

In this scheme a facade descriptor is considered regular, if it is composed entirely of regular surface-elements. Essentially the wall's of regular facades can be accurately clipped by removing only quadrilateral surface-elements. This means that the underlying clipping-algorithm only has to support axis-aligned (horizontal and vertical) splitting operations. The benefit of this is that determining which side of an edge a point/vertex is on (which is a integral task in constructive area operations) can be quickly determined using a conditional expression on a single component of the point/vertex. Based on this the large majority of architectural facades could be considered *regular* within this scheme. Figure 4.23 illustrates examples of 3D models of regular facades, from the City of Bath CAD model to clarify.

The important aspect is that such facades can be treated with a simplified clipping routine that trades off generality for efficiency. Furthermore since the only aperture shapes that need to be removed for such facades are rectangles, a number of topological enhancements are also applicable.

Note though that the categorisation of regularly composed facades does not restrict the form of the facade. It only applies to the objects that possess apertures which require *cutting-out*. For example in figure 4.24 note that under this scheme, the pillared structured (left) and the curved wall-descriptor (right) do not affect the classification of their respective facades (as composed of regular surface-elements), since they do not impact the requirements for aperture clipping. In essence, adornments and rail-able features (such as beams and columns) do not have to be transparent.

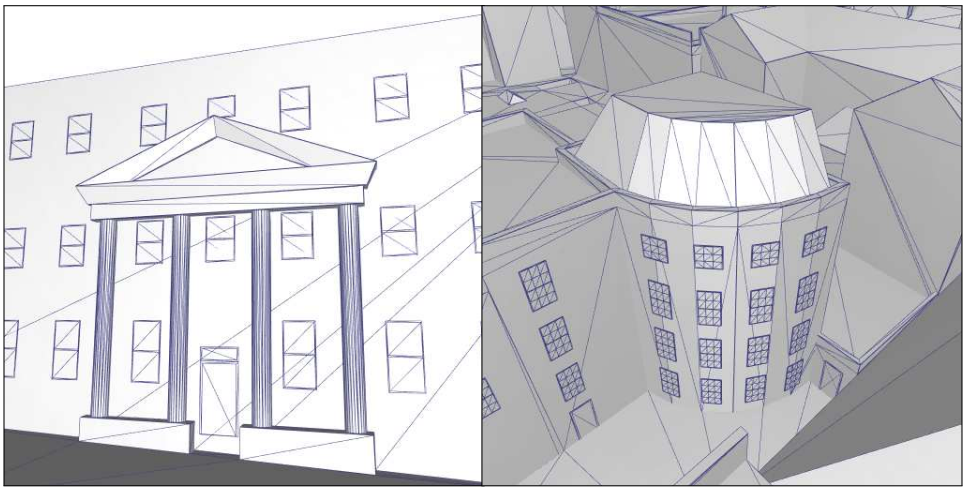

Figure 4.24: *Examples of facades composed entirely of rectilinear surface-elements with irregular facade features (from the City of Bath CAD model). Illustrating (left) a facade with additional surface features - the pillared structure, and (right) a curved facade modelled as set of planar pieces.*

Shortly this section discusses the enhancements suited to facades composed of regular surface-elements. For now (to keep the discussion flowing) facades composed of irregular and mixed type surface-elements are outlined.

Facades Composed of Irregular Surface-Elements

In this scheme, a facade descriptor is considered irregular, if it is composed entirely of irregular surface-elements. This means that the walls of irregular facades can only be accurately clipped by removing polygonal aperture shapes. Intuitively the quadrilateral sub-division of such walls (exploited by this scheme) helps to control the quality of the output mesh. However the critical point is that beyond this, there are few short-cuts that can be employed to speed up execution. Figure 4.25 provides an example.

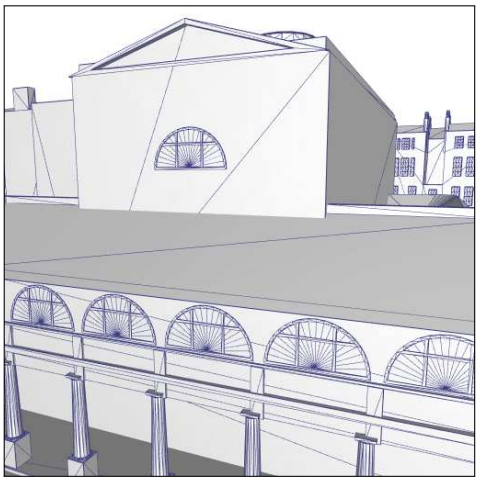

Figure 4.25: *an example of a facade composed entirely of irregular surface elements from the City of Bath CAD model: vitally the key difference between this particular example (and the previous facades which were defined as being regular in figure 4.23) is that each of its constituent surface-elements (that induce the requirement for a cut-out aperture) possesses an irregular (polygonal) as opposed to rectangular extremal boundary.*

Note that whilst the prevalence of facades composed of irregular surface-elements will vary from geographic region to region, in general the frequency with which they occur is less than that of *regular-facades*. Far more common, are facades composed of a mixture of regular and irregular surface-elements.

Facades Composed of Mixed Surface-Elements

Figure 4.26: *facades composed of a mix of surface-element types for two manually constructed building models in the City of Bath CAD dataset.*

In this scheme mixed-type facades are those composed of both regular and irreg-

ular surface-elements. Essentially if a facade contains rectangular and polygonal (such as arched or oval) apertures it is considered a mixed-type facade. Figures 4.26 and 4.27 illustrate examples of mixed-type facades.

Note that whilst the shapes of apertures vary in these examples they still obey the axis-aligned disjointness principle that was employed in the earlier dicing stage. The important aspect of mixed-type facades is that additional logic is required in order to determine how best to clip away each surface-element. This additional logic is simple and relies solely upon each surface-element's boundary. However mixing types in the clipping stage can restrict the enhancements applicable to the cut-out representation.

Figure 4.27: *further examples of mixed-type facades (which are composed of regular and irregular surface-elements) to complement those in figure 4.26: depicting (left) a supplementary pillared structure and (right) a curved wall whose descriptor is approximated with planar pieces.*

Practically the clipping method described so far will operate correctly for any type of facade in this scheme. However in principle (and indeed practice) it is quite desirable to take advantage of the restricted nature of the regular surface-elements in order to enhance the aperture wall model by reducing the number of vertices and faces in the output. The underlying idea in such cases is to reduce the amount of sub-dividing employed by adapting the clipping stage based on the surface-elements that are actually recovered.

Hence if a facade is composed entirely of regular elements, it does not make sense to treat it with a generalised polygonal difference operator, when a more efficient axis-aligned splitting routine will do. Figures 4.28, 4.29 and 4.30 seek to illustrate this notion. The key difference is that now (rather than sub-divide the wall without reference to the surface-elements present), the outcome will be directly controlled by the edges of each surface element.

In particular note that in figure 4.28 the outcome mesh contains far more vertices and faces than is desirable (given the regularity of the facade). This is because it does not take into account the position and extents of each aperture. Although the approach is appropriate for irregular surface-elements, for regular surface-elements, such behaviour could be considered wasteful.

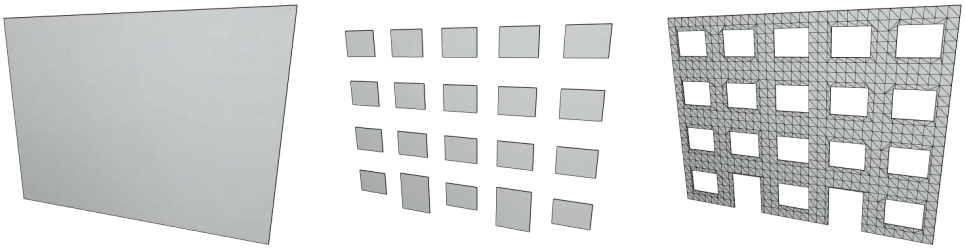

Figure 4.28: *an example of aperture clipping using the prescribed quadrilateral sub-division routine with fixed size pieces of 50cm x 50cm - the left-hand image depicts the wall boundary, the central image the surface-element boundaries and the right-hand image the quad-dominant wall mesh*

The enhanced aperture clipping method illustrated in figure 4.29 seeks to combat this. Note that (in the left-hand image) the edges of the facade's *grid-cells* coincide with the edges of the aperture boundaries. The simplification (right-hand image) simply merges the remaining neighbouring *grid-cells* first horizontally and then vertically to reduce vertex-count.

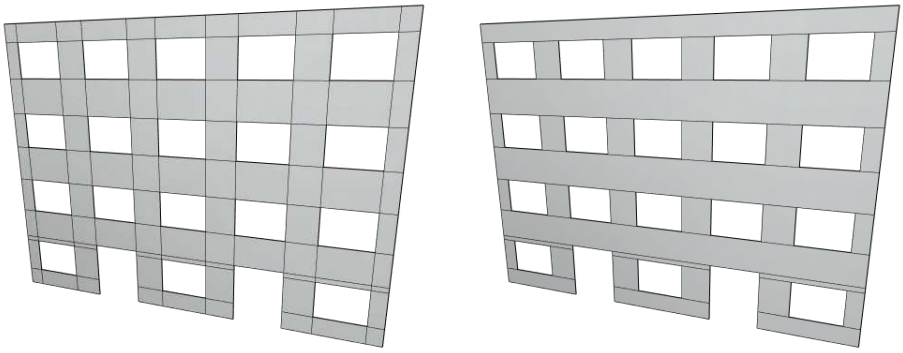

Figure 4.29: *an enhanced aperture clipping routine (for regular facades) applied to the example in figure 4.28: illustrating (left) the outcome wall-mesh by deriving clipping edges from the horizontal and vertical edges of each surface-element, and (right) the outcome of using a grid representation of the clipped facade to perform a cell-merging simplification technique.*

Remember the reason for the enhanced aperture clipping routine (specifically for regular facades) is to reduce the number of primitives used in wall meshing. To quantify this reduction: in the example the original aperture model (figure 4.28, right) possesses 2,632 vertices and 1,376 triangles. The surface-element guided clip (figure 4.29, left) possesses 440 vertices, 110 quads (220 triangles) and the cell-merged simplification (figure 4.29, right) possesses 136 vertices, 34 quads (68 triangles). This means the cell-merged edge-guided clipping used roughly 5% (68/2,632 $\approx$ 0.049 $\approx$ 1/20th) of the number of primitives relative to the raw quadrilateral clips. Obviously this ratio can vary greatly, depending on the quadding parameters used, however generally speaking, taking into account the edges of regular surface-elements during aperture clipping will significantly reduce the size of the output wall-mesh. Figure 4.30 illustrates a close-up of this difference.

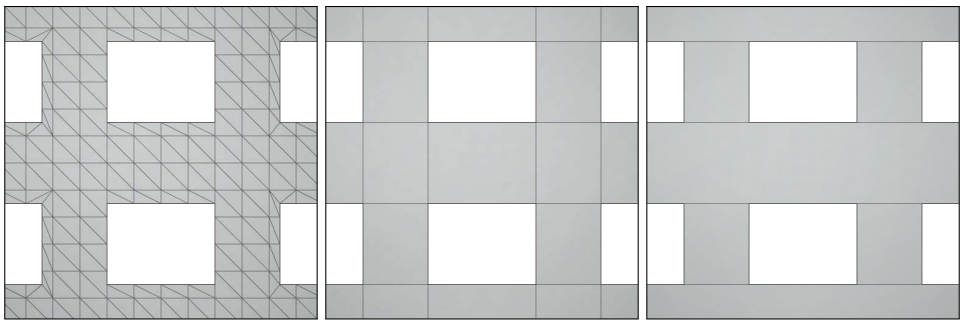

Figure 4.30: *close-up of the topology of the enhanced aperture clipping routine targeted at regular facades - illustrating (from left to right) the quad sub-division clips, the edge-guided clips and the cell-merged edge-guided clips.*

One critical point that should be noted relates to the application of this aperture clipping enhancement on *irregular* and *mixed-type* facades.

The key idea of the enhanced aperture clipper is to use the boundary edges of rectilinear surface-elements in order to automatically determine the optimal positions to clip the wall representation so as to minimise the number of faces in the resultant wall-mesh. Although we have discussed facades composed of regular surface-elements - in practice this clipping optimisation is also partially applicable to all facades that contain irregular surface-elements as long as they still obey the axis aligned disjointness principle.

You should now possess a clear understanding of what the ARROW algorithm is, why it exists and how it works. Before progressing to the results and analysis of ARROW the next section briefly outlines supplementary prior driven alterations that can be applied to the data-driven facade descriptors in order to enhance the semantic quality of the result. These enhancements represent extensions to the core method (slice, dice, rail and clip).

4.3.6 Snap (Optional)

ARROW (Accurate Railed Reconstruction of Openings and Walls) is a data-driven algorithm for facade reconstruction from unstructured laser-scans. The fact that it is data-driven leads to many positive behavioural features. However it also introduces some undesirable down-sides. This section discusses the priors and heuristics that may be employed to combat these.

In essence, the *snap* is an additional post-processing stage that optimises the arrangement of each facade according to architectural conventions and desires. In terms of why: the *snap* exists in order to support template and split-grammar based methods (that are used to address the problem of facade reconstruction) without impacting the core data-driven strategy.

This (optional) snap stage operates by identifying certain key patterns of regularity. These patterns include axis-aligned snap-lines, repeated surface-elements, groups of repeated surface-elements, reflectional symmetries and template matching common surface-elements split-logics.

Formally the input to the snap is an abstract descriptor of a facade: $F \leftarrow \{w, S\}$ - where F is the facade representation which is the tuple of w (the wall descriptor) and S the surface-element descriptors. The output of the snap is a modified version of the input facade-descriptor - that enhances the regularity of the representation: F' - such that given a unary regularity measure $r(a)$ that returns unsigned scalar values (with larger values corresponding to greater regularity) - then: $r(F) < r(F')$. Essentially this stage modifies the abstract descriptor of a facade (prior to mesh construction) in order to improve the visual properties of the output model - by maximising the return of a regularity measuring function. The regularity measure $r(a)$ can take many forms, but as an example - a simple measure that supports axis-aligned snapping is a weighted sum of the surface-element bounding edges that are coincident (colinear) with one another.

Remember the context of this is procedural reconstruction of architectural facades - and in particular shape arrangement and structural priors, symmetry and pattern detection. In terms of the relevance, this is an optional post-processing step that aims to behave in a similar manner to the non-linear model optimisation stage employed in airborne reconstruction. The key difference is that there is generally less call for this since one of the key expectations is a high-fidelity input scan.

4.3.7 Summary

The ARROW algorithm efficiently reconstructs sparse facade models from unstructured ground laser-scans. ARROW first *slices* the input facade into two subsets containing wall points and salient interest points. ARROW then *dices* the salient subset into disjoint connected components in order to isolate individual surface elements. ARROW then *rails* window and door aperture models in order to represent the salient facade features. Finally ARROW *clips* a polygonised wall using the boundaries of each surface-element in order to accurately represent apertures.

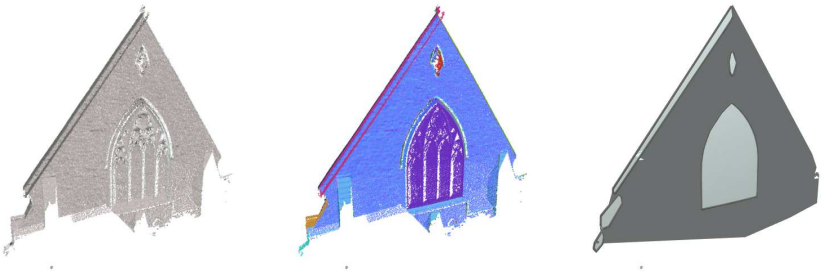

Figure 4.31: *illustrating ARROW → **A**ccurate **R**ailed **R**econstruction of **O**pen-ings and **W**alls: (left) input unstructured points, (middle) signed-distance-field split with segmented elements, (right) the output polygon-mesh (composed of a quad-dominant wall and railed surface-elements).*

The resulting facade model is composed of a quad-dominant wall mesh and individual window and door models (each constructed as a set of generalised cylinders). The key insights employed by ARROW are the use of verticality filtering to localise on the wall descriptor, planimetric projection as a means of dimension reduction, fast connected-component extraction based on axis-aligned disjointness and a formalism for data-driven modelling of arbitrary windows and doors (the surface-element). The advantages of the ARROW algorithm (relative to pre-existing facade reconstruction operators) include efficiency and the ability to construct high-quality semantically-meaningful data-driven facade geometry. Figure 4.32 summarises these stages visually with the example *irregular* facade used throughout this chapter.

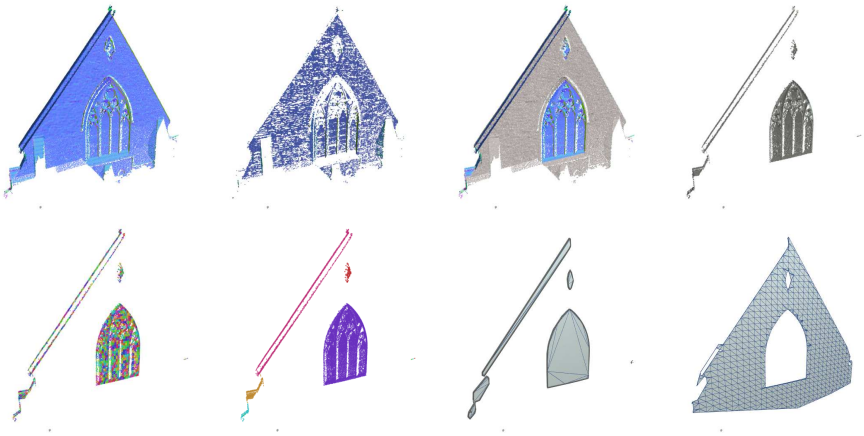

Figure 4.32: *summary of the intermediary data employed by the stages of the ARROW algorithm to transform unstructured facade scans into sparse (compact) facade models - illustrating (from left to right, top to bottom) the input facade scan shaded according to point-normals, the verticality filtered points used for wall localisation, the binary SDF-split, the subset of salient feature points, the KD-tree regions used to chunk the salient subset, the connected-components extracted from the salient subset's KD-tree, the railed surface-elements and the quad-dominant aperture wall model*

The next portion of this chapter documents the results of experimental tests and profiling of the ARROW algorithm on unstructured facade scans.

4.4 Experimental Results

This section documents the results of experiments that profiled the performance of the ARROW algorithm on unstructured facade point-clouds.

The results are structured in the following manner. First quantitative and qualitative results of reconstructing a set of test facades are documented. This stage by stage examination is followed by discussion of the comparative analysis of ARROW's surface element detection precision relative to human operators. Note that (unlike airborne reconstruction) certain parts of the ARROW algorithm cannot be evaluated independently of human supplied data. For example in order to measure the response of the window and door segmentation one first requires some form of ground truth facade representation that can be used to mark the results of ARROW's SDF-split and KD+AABB-connected component extraction. Essentially whilst one can measure the geometric error of each reconstructed facade model, (relative to the input point-cloud), in principle - in order to effectively measure the *semantic* accuracy - human labelled expected results are also required.

Once again the three key attributes that are profiled in these tests are the geometric accuracy of the facade models, their level of compression and the total execution time. Alongside these principal factors, the comparative analysis deals with the precision and recall of surface-element detection.

4.4.1 Synthetic Datasets

Prior to the concrete results - details of (and outcomes derived from) the synthetic datasets (used in controlled tests) are briefly outlined.

The synthetic facade scans were generated by sampling the surface of manually constructed facade models to simulate the behaviour of a fixed position radial scan. Simulated noise is applied in much the same way as for the synthetic airborne scans (discussed in the last chapter), i.e. the distance of each point to the virtual sensor was perturbed (to emulate sensor error) and a variable subset of the points are removed (to denote missing data and anomalous sensor readings). By running the ARROW algorithm on this data the following outcomes were observed:

- This implementation of ARROW only supports near-vertical planar facades - does not adapt to curvature or slanted walls without explicit support provided by a facade un-wrapping (linearisation/3D-to-2D-projection) operator.
- Slicing stage is stable when depth variance is present - but is less stable when nearly flush surface elements pervade. In particular the minimum depth variance required for robust surface-element segmentation (isolating windows and doors) was approximately $\pm$ 3-5 cm.
- Performance of chunk-based graph traversal (i.e. point-cluster grouping over point-level grouping) yields satisfactory (coherent) results efficiently whenever axis-aligned disjointness pervades a facade's surface-elements.
- Current implementation of SLADE handles window frame extraction (railing) reasonably well even as the amount of noise (signal error) increases. However its resolution of interior sash and pane divisions is fragile. In particular (even in noise-free tests) SLADE seems to perform better as the density of

the facade scan increases - and conversely degrades when the density (average inter-point spacing) of a facade scan decreases.

Note: the synthetic datasets were primarily used to support development and debugging of the conceptual stages employed by ARROW. In this sense they do not form part of the performance analysis on real-data - however they enabled the implementation of ARROW to be incrementally improved and refactored. Essentially the synthetic datasets allowed the conceptual limitations of ARROW to be analysed and addressed within controlled contexts - for which the expected outcome was known ahead of time. They are referred to here simply for completeness.

4.4.2 Roslyn Mews : Unstructured Facade Scans

The results of applying ARROW to unstructured facade scans are documented next. These results step through each of the processing stages employed (slice, dice, rail and clip) noting for each the outcome of ARROW's automatic processing.

Slice Results

Preliminary qualitative results of the initial slicing stage for a set of facade scans in the Roslyn Mews dataset are documented below. These results indicate the response of the data-driven signed-distance-field (SDF) heuristic that is used to divide each input facade scan into two subsets containing salient and wall-points.

Figure 4.33: *SDF-sliced unstructured facade point-clouds - illustrating the result of applying the initial automatic binary division strategy employed by the ARROW algorithm to real-world laser scans : the blue points correspond to the wall subset whilst the goldenrod points correspond to the salient subset - note: all facade scans are from the roslyn mews dataset.*

Figure 4.34: *continuation of qualitative results of SDF-sliced facade scans*

The qualitative outcomes of the slicing stage (in figures 4.33 and 4.44) indicate that the heuristic employed by the ARROW algorithm is relatively stable. Note that the slice behaves analogously for both regular and irregular facade types. However this is not to say that the implementation is perfect. Indeed the current slice relies heavily upon depth variance - which means that in instances of facade scans for which the variance in depth is insignificant then the slice has the potential to fail. The critical thing to note in these results is that it is not so much the presence of noise that controls the efficacy of this initial stage but specifically the geometric distance between salient points (corresponding to surface elements) and the wall-points. In other words the slice adapts well to noise, but requires depth variance.

Dice Results

Qualitative results of the outcome of the dicing stage are documented below. These results elucidate the performance of surface-element segmentation for the salient subsets of SDF sliced facade scans (generated by the preceding slice) - in order to clarify the performance of the detection strategy.

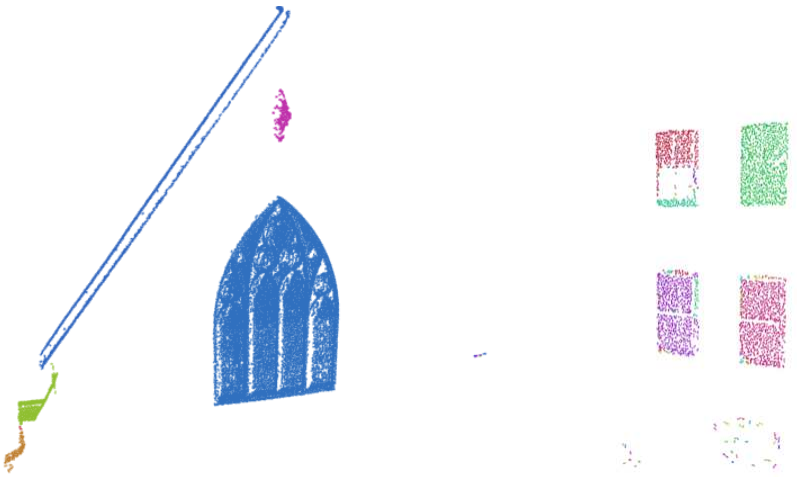

Figure 4.35: *surface-element segmentation results for the salient subsets of the SDF-sliced facade scans (constructed automatically based on clustering disjoint connected components from a KD-tree space partitioning) - illustrating the isolation of individual window point-clusters as a product of the axis-aligned disjointness principle employed by ARROW*

Figure 4.36: *continuation of segmentation results for the test facade scans*

The results of the dicing stage indicate that the key problem with ARROW's automatic surface-element segmentation strategy is under and over segmentation. Note in particular that the right hand facade (in figure 4.35) over-segments the top-

left window - resulting in two surface-element clusters instead of the desired one. One can see clearly that this result is largely due to the reliance on connectivity which (in this case) is not sufficient to group the cluster of points perfectly. However the remaining three windows for the facade are clustered correctly. Further more (in figure 4.36) note (in particular) the under-segmentation of the top set of surface-elements, which (despite being distinct) are clustered together as a single surface-element because they are not wholly disjoint. These results indicate the predominant problem with the data-driven nature of ARROW's dicing stage and are discussed in greater detail in the evaluation portion of this chapter.

Rail Results

Early stage results of the outcome of the railing stage are discussed below. These results are primarily preliminary in nature since the performance of the integral SLADE algorithm still requires refinement in order to better deal with irregular surface-element point-clusters. These results also document some of the key issues that prevent stable extraction of split-logic axes in the presence of irregular surface-elements.

Figures 4.37 and 4.38 document qualitative results of the rail stage. Each window frame is the result of SLADE's boundary extraction process. Note though the omission of sash components to represent the structural division of panes. In essence the current version of SLADE can only robustly extract frame components. From a practical perspective the frame shapes are useful for analytic tasks (for example building energy efficiency simulation). However without the interior sash-divisions they are less suitable for photo-real visualisation.

Essentially this research observed that although (in principle) the extraction of split logic axes is feasible and indeed quite practical for regular (rectilinear) surface elements - the formulation currently used is not sufficient for the recovery of irregular split-logic descriptors. In particular the approximation of split-logic axes containing curved components cannot be adequately handled using piecewise linear approximations - and (based on the experiments) seems to mandate the use of circular arcs and bezier curves in order to facilitate a correct representation.

Clip Results

Qualitative results of the final clipping stage are documented below for the test facade scans used throughout this section.

Figure 4.37: *results of ARROW's quad-dominant aperture clipped wall meshing stage for the test facades examined in this section - illustrating planar cut-out wall models and the railed surface element boundaries - generated automatically by the algorithm*

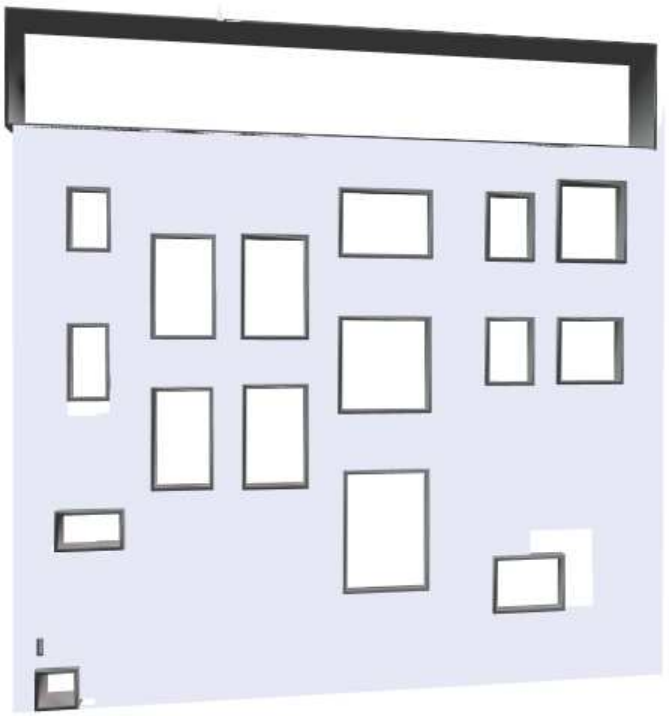

Figure 4.38: *continuation of aperture clipping wall modelling results*

These results indicate that the clip represents an (almost entirely) infallible component of the ARROW algorithm in the sense that it will generally never alter the error attributes of the resulting facade models. Ultimately the attributes of the clip only impact the typical surface area of faces in the quad dominant wall mesh. In this sense the clip can be considered the most stable stage of ARROW. Indeed the power of the clipping strategy employed is most apparent when applied generically to forward chaining facade modelling problems - wherein one will notice that the clipper optimised for regular surface-elements, not only minimises the number of vertices and faces used but is also often suited to irregular and mixed-type facades that obey the axis aligned disjointness principle.

However these results also indicate how crucial surface-element extraction is to the process of facade modelling. In particular note that in the right-hand facade (illustrated in figure 4.37) and the larger regular facade (in figure 4.38) errors with surface-element boundary resolution lead to window apertures that deviate from one's expectations. In figure 4.37 this is a result of the previous over-segmentation, whilst in figure 4.38 the problem stems from both under-segmentation and issues in the boundary extraction sub-process of the preceding railing stage. So whilst the logic of producing high-quality aperture wall meshes automatically is reasonably robust - in practice, any errors from the previous modelling stages will have an impact on the correctness of the final facade model. Vitally be aware that these automatic aperture clipped results were produced without the imposition of the *snap-logic* priors and heuristics discussed in section 4.3.6 - and effectively document the purely data-driven result of the ARROW algorithm.

Quantitative Measures

Quantitative results from these experimental tests of ARROW on real-world unstructured facade laser-scans are outlined in the table in figure 4.40.

Facade	Input Points	Runtime	Output Vertices	Output Triangles
church_face	129,897	3.768 s	6,828	3,518
house	25,436	0.744 s	3,283	2,637
multi_storey	119,906	5.912 s	2,765	2,082

Figure 4.40: quantitative measures of ARROW's performance on the unstructured scans of facades in the Roslyn Mews dataset to complement the qualitative results

4.4.3 Comparison with Human Operators

This portion of the results discusses ARROW's performance compared to human operators. The aim is to clarify the precision and recall of the window detection process based on the proposed methodology. As eluded to earlier - comparison with manually produced ground-truth data plays a vital role in understanding how well suited the ARROW algorithm is to addressing the task of automatic facade reconstruction from unstructured laser-scans, because ultimately there is no intrinsic relationship between the geometric results and that which one could consider as the semantic correctness of each facade model. Essentially - although each outcome may look adequate, it is impossible to tell whether the results model coherent representations of each facade without some pre-existing notion of what constitutes a coherent representation. This aspect of the performance analysis renders the problem of facade reconstruction akin to that of classification (addressed in

chapter two) and distinct from airborne reconstruction (chapter three).

In this research determination of surface-element recall and precision is measured in two dimensions using planimetric projections of each facade scan - i.e. image projections with corresponding 2D vector shapes. An important alternative is full-blown three dimensional comparison relative to manually constructed facade models. As such there are inherent limitations to the comparative strategy employed - which are discussed in greater detail in the analysis and evaluation portions of this chapter. However the key benefits of this restricted form of comparison include efficiency, simplicity and (as a result) the ability for non-technical users (those lacking professional 3D modelling experience) to produce ground-truth data that can be used to determine if ARROW correctly detects all windows and doors.

During the experiments 28 individuals (employees of the company that funded this research) were asked to manually label window apertures on ortho-aligned projection images derived from the testing facade point-clouds used in this chapter.

From this the overall recall rate for surface-elements in the testing facade scans was calculated as product of testing for intersection between the manually plotted and automatically generated surface element bounding boxes - and roughly fell in the range of 60%-70% (relative to the participant's labels). Although the precision of the detected surface elements was slightly higher at approximately 70%-80% - it is clear that the current implementation does not yield industrially satisfactory results (from the perspective of surpassing a humans perceptual performance) - for which the expectation is precision and recall rates upwards of 95%. Essentially although the results are promising - they are not yet at an industrial standard which would enable fully automatic exploitation of ARROW - and hence a technician would likely still be required to vet, validate and correct ARROW's operation.

Note: though that despite the requirement to improve ARROW's precision and recall - profiling revealed a tight geometric fit for the correctly isolated surface-elements - relative to the human-data. This suggests that by improving the *dicing* stage's performance - one can increase the surface-element detection accuracy of the ARROW - yielding a more robust operator - without necessarily having to alter the subsequent *railing* and *clipping* stages.

4.5 Analysis and Evaluation

This section evaluates ARROW's performance by comparing the results and outcomes to the behavioural objectives set out in this chapter's introduction. It provides high-level analysis of the extent to which ARROW is effective.

Facade Segmentation

The results indicate that ARROW's slicing stage responds reasonably well in the presence of sensing noise. However the underlying ideology relies upon depth variance - which means practically that scans captured from large distances (especially those that fail to capture significant depth-variance) are less suited to being treated with the signed-distance-field split.

ARROW's dicing also responds reasonably well - again even in the presence of significant sensing noise and missing (or partial) data. However the critical issue

with the dicing stage is over and under segmentation - which results from the data-driven nature of the connected-component extraction.

There is currently no additional facility built into ARROW to mitigate this (save altering the connected component distance and maximum KD-tree depth). The problem is that as one increases the maximum tree depth (or reduces the CC-distance) over-segmentation typically increases whilst the inverse action tends to result in under-segmentation. Striking a good balance can be a tricky process, and so one viable solution may be to address this balancing act in a similar manner to MAMMAL's MARS algorithm - i.e. an initial over segmentation followed by a subsequent non-conformal suppression. However the challenge lies in the fact that one cannot use surface-area to determine conformance (as in MARS) and would require an alternative method of determining clusters that require merging.

Surface-Element Recovery

Boundary extraction is shown to be stable for both regular and irregular surface-elements. However full-blown three-dimensional comparative analysis with human produced facade models would be useful in terms of yielding greater insight into the precise error associated with aperture localisation.

Split-logic resolution on the other hand still exhibits significant problems - especially in the presence of irregular interior sash divisions. Although regular (rectilinearly based) split-logic axes can be extracted robustly, practically the performance of SLADE on irregular ecclesiastic windows (such as the church face) is unsatisfactory. This is particular true for the case of split-logic axes exhibiting curvature - for which a possible solution is to refactor the piece-wise linear approximations in order to take into account circular arcs and bezier curves. This is a key area for continued investigation.

Computational Efficiency

Positively the execution time of ARROW places it at the faster end of the architectural reconstructive spectrum. In particular the results of reconstructing the test facades demonstrate ARROW's ability to produce sparse geometric representations within seconds - a feat that is currently unmatched by the existing methodologies (be they point-cloud driven or image-based).

Whilst speedy execution is important (since it will affect the number of algorithmic iterations one can make given a fixed period of time), once again it can be considered secondary to the crucial unresolved geometric problem of recovering sash descriptors for irregular surface-elements.

Qualitative Analysis

Finally in considering the qualitative properties of the facade models recovered by ARROW - it is clear that the exploitation of an explicit (albeit heuristic) modelling strategy (as opposed to an indirect sampling approach or dense surface reconstructive approach) results in the ability to produce architectural models with far greater semantic meaning than otherwise. Although the results are not perfect - it is clear (at a glance) that each result models a structured facade. Due to the

fact that ARROW isolates wall-points from salient points in a manner that respects arbitrarily sized windows and doors, there are fewer restrictions on the extents of surface elements, which has the added benefit of ensuring that details at varying spatial scales are preserved. Figure 4.39 illustrates this with a close-up of the aperture clipped wall-mesh of the church facade used in this chapter.

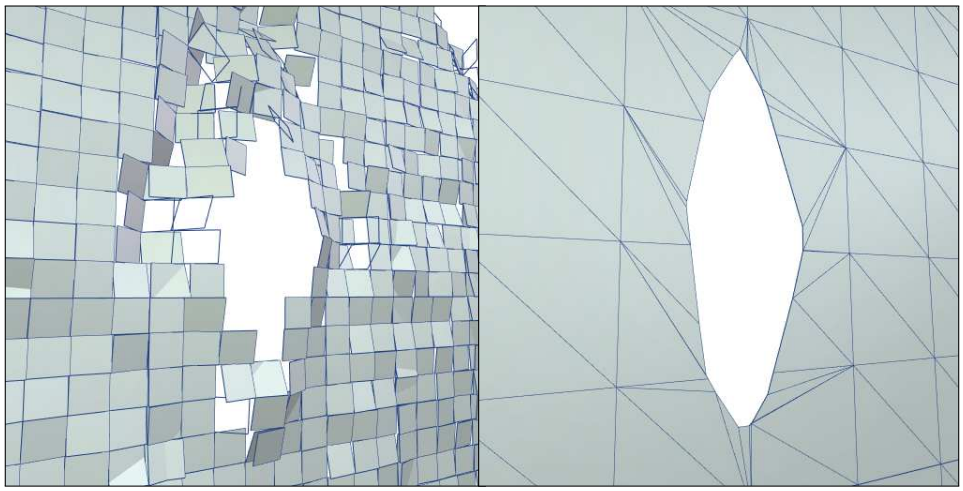

Figure 4.39: *comparing the topology of volumetric reconstruction using the regular arrangement of planes (left) to the ARROW algorithm (right)*

Note that even without the enhanced clipping or symmetry based revision, the aperture model clearly delineates the extremal boundary of the surface-element in question. Essentially although ARROW executes in a similar time-frame to data-driven facade reconstruction methods, it yields higher quality facade models.

In evaluating the results it is clear, that although the current implementation has certain limitations, the underlying ideology possesses crucial benefits relative to pre-existing approaches to automatic facade reconstruction. In particular is serves the driving aim of marrying the computational efficiency and geometric precision of data-driven reconstruction methods with the higher-geometric quality of templated (or library-based) strategies. In this sense it is fair to say that ARROW partially meets the objectives set out in this chapter's introduction.

Now (as briefly mentioned), although the idea may be sound, the implementations require improvement. The next portion of this chapter discusses these critical areas requiring continued development as a sign-post for future researchers charged with coordinating fast, accurate and sparse automatic facade reconstruction from unstructured point-clouds.

4.6 Enhancements and Improvements

This part of the chapter deals with the future investigative areas and potential advancements to the ARROW algorithm. The key concern is documenting the remaining theoretical problems and the aspects of the implementation that could be revised to enhance ARROW's performance. These changes and further research topics are arranged into groups which each group corresponding to a specific evaluative attribute. These attributes correspond to: geometric accuracy, reconstructive robustness, computationally efficiency and performance analysis.

Geometric Accuracy

In order to improve the geometric accuracy of the facade models produced by ARROW, two main areas of investigation are proposed. Loosely speaking they aim to 1) improve the accuracy of each surface-element's model and 2) add support for additional facade features and adornments that are also amenable to modelling with generalised cylinders such as plumbing pipes, and beams and columns.

- **Accumulated Sweep Profiles** - the basic idea is to actively mine the *profile* shapes that are swept about each surface-elements frame and pane polygons, by accumulating flattened slices of points along the boundary of each polygon - in a similar manner to [151] only applied at the level of individual windows and doors rather than entire facades. This has the potential to improve the fit of each surface-element model relative to using default box and bevel-box profiles that merely approximate the thickness of each sweep.
- **Adornments and Surface Features** - here the aim is to ensure that features of a facade that may be omitted by segmentation (due to the fact that they do not represent significant or aperture-like clusters) are modelling using appropriate generative functions. As mentioned the key examples are pipes, beams, columns, in addition to electrical fittings and aesthetic decorations. Although these features add to the physical accuracy of a facade model, the key issue becomes modelling arbitrary adornments without resorting to dense surface reconstruction. For pipes, columns and beams, this problem is addressable - since their regularity allows for generalised-cylinder based approximation, however in the case of irregular features (such as gargoyles and similar non-structural aesthetic adornments), the task is less simple.

Reconstructive Robustness

These alterations seek to improve the response of ARROW in the presence of low-density input scans and in terms of its ability to recover irregular surface-element split-logic descriptors. In other words these improvements are intended to make ARROW more robust and versatile.

- **Split-Logic Resolution** - foremost the SLADE algorithm needs to be refactored to take into account circular arcs and bezier curves as opposed to simply piecewise linear edges - however addressing this (even in two-dimensions) without randomly sampling has proven itself a wholly non-trivial task. Further

though the method of localising on individual *split-edges* prior to arrangement should (ideally) perform non-maximal suppression in concert with reflectional symmetry detection and hyper-graph based edge-connectivity refinement (similar in-vein to [37] - only without the random traversal strategy). The non-maximal suppression would address the need to filter anomalous data-driven edges, whilst the symmetry detection could be safely applied locally without altering the overall arrangement of the facade (and hence the error profile). The hyper-graph edge-snap would provide a parametric means to control the scope for alteration to the filtered split-edges during the refinement process.

- **Reconstructing Curvature** - this is to add facade-unwrapping support to address curved facades (beyond surface-elements exhibiting curvature). This calls for three key components. Firstly one requires a function that extracts non-linear planimetric facade-descriptors from unstructured facade-scans. Secondly one requires a function the performs a non-linear projection of a set of three-dimensional points onto an arbitrary but consistent plane such that the curved facade descriptor is represented as a 2D planimetric set. Thirdly one requires a function to perform the inverse projection to a set of model vertices in order to restore the ARROW facade mesh from planimetric to world-space. The critical aspect of this alteration is coordinating the curved facade descriptor - because one also has to detect and maintain the set of key-points (corner-positions) during the non-linear projection.

- **Low-Quality Scans** - this alteration seeks to address the fact that ARROW demands reasonably high-density scans - because it relies largely on depth variance. Two changes are suggested to combat this and improve ARROW's efficacy. Firstly to the segmentation method and secondly to the SLADE algorithm. In terms of the segmentation the obvious change is to deviate from a purely geometric strategy and introduce heuristics. For example exploiting prior knowledge about the fact that the reflectivity of glass surfaces perturbs the return of a scanner could be used to guide ARROW to hole regions within facade scans in the case of data surveyed from larger distances. In terms of SLADE the useful alteration for lower-quality scans would be non-stochastic template fitting instead of data-driven extraction. This would certainly improve the recall - however the precision has the potential to suffer since the templates recovered would be at best loose approximates.

- **Missing Data and Partial Scans** - although this aspect could certainly be improved - the simple reality is that there is no logical way to fill in significant holes and regions of missing data without synthesising data. This means that in order to hole-fill one has to make up data in one manner or another. Whilst in image-processing and computer vision this is less of a problem because often the result is judged qualitatively - in active sensing this can have undesirable repercussions. Foremost hole-filling gives a sense of completeness to a reconstructed result that is actually false - and introduces greater geometric error relative to the input point-cloud given that faces in the output are not actually represented in the input - reducing the overall correspondence.

Computational Efficiency

In order to improve the computational efficiency of ARROW, one type of alteration is discussed - parallel execution paths or simultaneous multi-processing.

- **Parallel Reconstruction** - involves distributing ARROW's work-load across multiple simultaneous execution paths. This change relies on the fact that once top-level facade isolation is complete, each facade and its surface-elements can be modelled independently. This research has experimented with basic CPU-bound parallelism, however further investigation should also profile the performance of GPU-bound multi-processing.

Performance Analysis

These changes address the limitations of ARROW's current (predominantly 2D) performance analysis methods - with the aim of improving the accuracy of the error-measures used to determine recall and precision.

- **3D Human Ground-Truth Data** - this requires deviating from the planimetric plotting strategy (currently employed to enable non-technical users to produce ground-truth facade-descriptors) in favour of a simple drag-and-drop 3D facade editor that would allow technicians to produce ground-truth facade models from each point-set. Note: the high-level trade-off in this revision is increase evaluative accuracy - relative to greater time requirements for ground-truth production.
- **Greater Range of Test Scans** - so as to enable rigorous evaluation of ARROW's performance across a greater variety of facades. This calls for a wider selection of unstructured architectural facade scans - ideally captured from a range of varying distances and with varying density and error profiles.
- **Comparative Analysis Relative to State-of-the-Art** - this investigative track seeks to combat the primary expositional short-coming of the ARROW algorithm to date - by profiling its performance directly against pre-existing facade-scan reconstruction methods - both dense and sparse. This is an important improvement from the perspective of gaining academic credibility by quantifying ARROW's performance relative to prior methods - however it is important to note that such exposition should be considered a secondary concern relative to the resolution of the technical problems that remain.

4.7 Discussion and Summary

This chapter presented a simple yet reasonably effective algorithm for reconstructing sparse facade models from unstructured laser scanned building point-clouds. As part of this the chapter introduced a novel abstract representation for windows and doors, named the surface-element. The surface-element is a versatile *procedural-primitive* for algorithmic window and door modelling. It is essentially a function that generates 3D geometry from 2D shape input. The power of the representation is the layer of abstraction it injects between the structural definition of a window or door and its realisation as a concrete 3D geometric model. The abstraction enables the ARROW algorithm to automatically construct accurate high-quality

data-driven aperture models that reflect the semantics of each cluster of window and door points. This chapter also documented the results of experimental tests of ARROW using a number of unstructured ground facade laser scans - for each considering the quantitative and qualitative performance alongside comparative measures relative to human CAD technicians.

Whilst these preliminary results are promising in terms of ARROW's computational efficiency and the compactness of the facade models it produced, this chapter identified the resolution of surface-element split-logic axes as the vital area requiring further research and development. Though ARROW's SLADE algorithm deals well with facades composed of *regular* surface-elements, its performance on *irregular* surface elements demands more attention. In particular although the boundary extraction and frame railing is stable - resolution of the characterising division of space is still lacking in terms of robustness given ecclesiastic windows (such as the church-face referenced throughout this chapter).

Despite this early-stage qualitative comparisons to sparse reconstruction algorithms demonstrate the benefits of ARROW in terms of structural accuracy, compactification, and the semantic richness of the aperture-clipped models.

So whilst the ideas underlying ARROW are sound in principle and are shown to be viable (feasible), in practice more work is required to handle *irregular* surface-elements and deal with robustness issues that may present as the product of low-quality and/or low-density input facade point-clouds.

Finally, in order to wrap up this chapter's discussion of fast, accurate and sparse facade reconstruction, the key principles, technical-developments and evaluative outcomes are revised in the enumeration following:

- **Signed-Distance-Field Binary Point-Set Segmentation** as an intuitive data-driven method of quickly isolating salient clusters of point in unstructured laser scans of architectural facades - that vitally can isolate both regular and irregular window and door apertures.
- **KD-Tree + AABB Spatial Optimisation** for Efficient Connected Component Extraction from SDF-sliced facade scans. This is predicated on the observation of axis-aligned disjointness that typically pervades the arrangement of surface-elements upon facades.
- **The 'Surface-Element'** - a Procedural Primitive for Automatically Generating High-Quality, Data-Driven 3D Window and Door Models.
- **Split Logic Axes Detection and Extraction** as the integral component of ARROW - that, given a cluster of points (that correspond to a surface-element) extracts the principal axes which encode the structural division of the interior sash geometry of the window or door.
- **Polygon-Clipping Quad-Dominant Aperture Wall Meshing** - and associated extensions for minimising vertex count in regular facades by deriving split edges directly from the extracted surface-element boundaries.
- **Extendable with Pattern-Detection Techniques** such as axis-aligned regularisation, repeated-element detection, symmetry detection, group-detection and template-driven (model-based) methodologies.

Chapter 5

Further Research and Development: Merging Airborne and Ground for Automatic Temporal Updates and Future Investigations

This penultimate chapter discusses the work in progress and future topics of investigation related to data-driven procedural modelling of the built environment.

The preceding three chapters documented fresh approaches to tackling fundamental problems in architectural reconstruction from point-clouds. The developments represent stand-alone algorithms for semantic error-detection, 2.5D mass reconstruction and 3D facade reconstruction.

The driving aim of this thesis is enabling automatic continuous temporal updates driven by the concept of selective-reconstruction (i.e. reconstructive culling, lazy reconstruction or change-aware reconstruction). Hence this final chapter considers the amalgamation of these algorithms in the pursuit of this aim.

This chapter also discusses the key remaining open problems that require addressing through continued research - and outlines directions for future investigation related to automatic AEC asset creation from laser-scan data.

This chapter's content is structured as follows:

- The Merging Airborne and Ground section outlines the structure of a multi-modal reconstruction algorithm CUBE (complete, unified building exteriors).
- The Automatic Temporal Updates section discusses the process of effecting automatic continuous temporal update to city-scale architectural CAD models driven by airborne and ground laser scanned point clouds.
- The Remaining Open Problems section documents the issues and limitations that must be resolved to improve the performance of the reconstructive techniques proposed by this research in support of automatic temporal updates.
- The Further Investigations section outlines future topics for investigation.

5.1 Merging Airborne and Ground

unifying mass & surface...

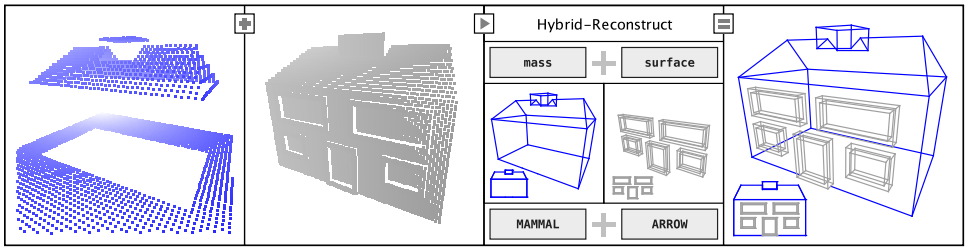

Figure 5.1: *an overview of the key stages in multi-modal reconstruction - illustrating from left to right: (blue) input airborne point-cloud, (gray) input ground point-cloud, components reconstructed from the input, and (blue and gray) the resultant building exterior model.*

5.1.1 Overview

This section tends to the problem of automatically reconstructing complete building exterior models using airborne and ground based laser-scanned point clouds. The core idea is to merge the massing models recovered form the airborne scans with the facade models recovered from the ground based scans. This section outlines a multi-modal merging operator called CUBE (complete unified building exteriors), which seeks to address this problem. The aim of the CUBE operator is to provide a fully automatic means for amalgamating different types of laser scan into coherent building models. The context of the chapter is again automatic architectural model reconstruction from laser-scanned point clouds. However now the focus is merging different types of scan data in order to yield complete building exteriors. Fundamentally this is not a new problem, however pre-existing approaches to addressing it are plagued by a number of key limitations. The most detrimental being poor computational performance, limited geometric accuracy (especially in the case of exploiting image-data) and inadequate geometric model quality.

The core idea here is to efficiently guide the segmentation of large unstructured ground scans using wall positions recovered from aerial massing models. In essence rather than rely on exhaustive traversal of the ground scan - the CUBE operator uses MAMMAL's reconstructed airborne models to quickly determine the salient regions of the unstructured scan, isolates individual facade point-clusters and feeds them into ARROW. For reference the key objectives, requirements and desired behavioural characteristics of the operator are itemised following.

- Geometric Accuracy in the sense each merged building model is correct relative to the input points and effectively characterises the building.
- Computational Efficiency such that city scale sites can be robustly merged in a reasonable amount of time on commodity hardware.
- Out-of-Core Execution in order to ensure that mid-large unstructured point-clouds can be processed without memory based restrictions.

- Compact, Semantized Exterior Models that largely obey the MAMP principle and are suitable for interactive rendering and simulation.
- Suitability for City-Scale Virtual Environments especially utility for realtime planning and visualisation tasks coordinated with HMDs.
- Robust to Sensing Noise and Geometric Degeneracy such that artefacts and anomalies (such as ghost/shadow points or speckle noise) do not degrade the quality of the building models that are produced.
- Robust to Missing and Partial Data - in particular the expectation of graceful degradation in the presence of low or null density regions in the unstructured ground scan - and the (fallback) ability to mine facades directly from the input ground scans when regions with little or no data encumber the recovery of massing models from airborne scans.

The outline following is a breakdown of the organisation of this section.

- The background and context section discusses the key related research.
- The methodology portion outlines the CUBE algorithm in terms of its constituent processing stages (which are: registration, mass-reconstruction, facade-reconstruction and then model-unification).
- The preliminary results portion outlines qualitative outcomes from CUBE's development to date. Note: these represent work in progress.
- The discussions and summary portion concludes this outline of CUBE.

5.1.2 Background and Context

Before outlining the methodology employed by CUBE - the relevant prior research is briefly recapped. The aim is to revise alternative methods of addressing the multi-modal reconstruction problem from airborne and ground point-clouds.

Key Related Work

Recall that Calberg et al. [12] propose an efficient multi-modal surface reconstruction operator - for airborne and ground point-clouds. Their system scales well - however it yields dense surfaces - making it less suitable for building reconstruction. This is an example of a data-driven analytic method. Its strength is its efficiency and its support for texturing the surfaces - however its limitation is its lower model quality relative to sparse operators. It also yields little semantic information about each architectural component - i.e. it does not isolate windows and doors. Still the key aspect is that it demonstrates that it is feasible to merge high-resolution unstructured ground scans with low-resolution airborne scans at scale.

Note: whilst many pre-existing works deal with the constituent problems (registration [40][30][146], airborne reconstruction [163][165][73] and ground reconstruction [70][5][23]) in isolation, comparatively few tend to the multi-modal merging of top-down and street-level models. For example whilst Lin et al. [75] construct models with roof and facade detail - the inputs to their system are ground-scans.

Indeed (to the author's knowledge) there are currently no multi-modal building reconstruction operators that are able to construct sparse semantized models from

airborne and ground laser-scans automatically.

To recap: whilst methods for dense textured surface reconstruction from airborne and ground scans are reasonably well established, approaches for data-driven sparse reconstruction are still lacking in terms of their computational efficiency, and the semantic and structural quality of the models they generate.

5.1.3 Methodology

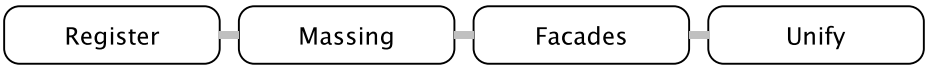

Figure 5.2: *Key Stages of the **CUBE** Algorithm*
→ Complete, Unified Building Exteriors

Figure 5.2 outlines the key processing stages in CUBE which roughly involve:

1. Registration of the structured aerial and unstructured ground laser-scans in order to determine a transformation that aligns them.
2. Aerial Reconstruction by exploiting the MAMMAL algorithm in order to construct sparse roof massing-models from the airborne scan data.
3. Ground Reconstruction based on guided segmentation of the ground scan and the exploitation of the ARROW algorithm in order to construct sparse facade models for each building model recovered.
4. Unification of the aerial geometry with the ground geometry in order to produce complete building shells (roof + facade details) by merging and refactoring the two distinct types of architectural geometry recovered by the MAMMAL and ARROW algorithms.

Register: Airborne and Ground Scan

CUBE's registration stage is an image-based top-down (planimetric) method of aligning the input unstructured ground scan with the corresponding input structured aerial scans. It provides an intiial aligning transformation that maps the ground scan data to the airborne scan data. It exploits a density based representation of the ground data in order to estimate facade regions that are aligned to wall positions from the airborne scan data.

The input is a structured airborne scan and a corresponding unstructured ground scan of the urban region being reconstructed. The output is a rigid-body transformation that maps the ground scan to the aerial scan.

The context is feature-based point-cloud registration. In particular the alignment of partially corresponding multi-modal architectural point-sets. This is a special case of point-to-point registration since the inputs actually represent completely different perspectives of the same underlying object. This means that the correspondence between the point-clouds is actually quite low - because the important features captured by each representation are typically not present in the alternate. As such one requires a method that takes into account the fact that the airborne scans only represent roof details while the ground scans largely represent facade detail. This

means that a point-level error-minimising registration routine such as ICP will not work directly. The challenge instead is to automatically identify an effective characterising feature-descriptor for the input scans that mitigates this.

The relevance of this initial alignment is a product of it being the inaugural compenent of CUBE. It is responsible for ensuring that efficient guided segmentation of the ground scan is feasible. Without stable registration, there is no way to relate the two datasets and each would have to be treated independantly, which would significantly increase the execution time of the ground reconstruction. In essence registration enables the subsequent fast association of on-plan facade walls and clusters of unstructured points. The utility of this is its efficiency via its means of dimension reduction and its stability given the weak correspondence between the two datasets. The core idea is that walls are the only constant between the two types of input scan. They can be efficiently extracted from both representations (by verticality) and are easier to align than organic features or roof-ridges or facade-elements (which will be partially or completely omitted).

The biggest limitation of this is the expectation that wall positions in the ground scan correspond to wall positions in the aerial scan. Given that they act as the shared frame of reference between the two types of input, one requires that the accuracy of the airborne walls is sufficiently high else the alignment can fail. Two examples of when this may not be the case are when temporally varying scans are supplied as input, and when non-parallax features such as tunnels and underpasses are not captured in the airborne scan. Another limitation is that the density mapping feature extraction strategy is not a perfect solution, and in particular the transition from the structured 2D density map to wall edges is somewhat of an ill defined problem. This meant that an amount of trial and error was required to identify a reliable edge filtering strategy. Fortunately the previously defined algorithm GRAILS comes in very handy for this. However further work is still required to improve the recall of walls recovered directly from the unstructured ground scan based on the density heuristic.

Fundamentally this proved quite a tricky problem because point-to-point methods that rely on there being a high correspondance between the registrees tend to yield nonsense for airborne and ground registration. The method that proved quite robust was to supply an initial rough alignment and then minimise wall-point-to-wall-point error between the two. However this is only feasible if user-guided registration is allowable and detection of wall points is stable. Attempts to compute a good initial alignment (even stochastically) were dissapointing especially in instances where the deviance between the inputs involved a rotary component in the transformation model. This is still an on going problem and at present the best option in terms of stability is to perform the feature based registration.

A key enhancement would be to propose a multi-modal scan correspondance operator that understands the difference between airborne and ground based scans, such that missing roof or facade data in one representation or the other does not prevent the measurement of similarity. However the logic of such a comparator is by no means trivial. The likely future investigative track is to improve the performance of the feature based registration. The reason that this is a difficult problem is that although it looks quite simple from the perspective of a human, in practice there is simply very little numerical relationship between what we perceive as an aligning transform and the transformed position of the registree points. The feature

based registration proved the only feasible means since although the sampling will vary greatly one can expect walls to be present in both types of scan. Beyond this there is little to go on in terms of minimising point-level registration error.

This difficultly also manifests as invalid alignments when exploiting mutual information. The critical point is that this is not simply a multi-modal problem. It is a weak correspondence multi-modal problem.

To revise, CUBE's initial registration aligns the input airborne and ground-based laser-scans, based on minimising the error between derived wall-location feature descriptor maps. Once registration is complete, CUBE then proceeds to reconstruct airborne massing models - this is discussed next.

Massing: Reconstruct Airborne Models

CUBE's airborne stage is an automatic 2.5D mass model reconstruction stage that is driven by the previously defined MAMMAL algorithm. CUBE exploits MAMMAL in order to turn the airborne scan data into clean, compact and accurate building masses that form the basis of the urban scene being reconstructed. To achieve this, CUBE simply invokes MAMMAL on the input aerial scans, which segments, vectorises, projects and optimises 2.5D mass-models that are then added to the return scene-graph. The inputs to this stage are the structured airborne scans (digital surface and terrain models) and the set of algorithmic control arguments used by MAMMAL. The output is a set of objects of type *MassModel* - with each mass model being composed of a wall mesh and a roof mesh.

This is a vital component of CUBE, because it provides the basis for complete building reconstruction. Abstractly one can think of the mass-models recovered by MAMMAL as analogous to a pizza-base, upon which the character giving facade surface detail models (recovered by ARROW) are sprinkled as the *toppings*. Further the geometry returned by MAMMAL is crucial to accelerated reconstruction of the ground scans, because it acts as a means to quickly isolate regions with high probability of containing salient facade points. Without CUBE's use of MAMMAL, there is no efficient means to filter out clutter and non architectural points - which means an exhaustive traversal of the ground scan would be required.

CUBE's use of MAMMAL is predicated on the notion that if CUBE has a good idea where buildings are in a dataset, then it can greatly speed up the efficiency of the reconstruction of ground scan data by using the knowledge to selectively model only salient regions. Hence by first reconstructing the airborne laser-scans, CUBE produces *guiding* massing models which significantly reduce the workload in recovering building facade detail. Essentially MAMMAL provides a base-scene-graph to which detail is appended using ARROW. The key insight is that knowing where walls occur improves CUBE's efficiency. Moreover since MAMMAL implicitly yields such information - its execution prior to ground reconstruction serves to optimise the subsequent processes. The advantage of this is that CUBE ends up killing two birds with one stone. It simultaneously addresses the problem of automatically modelling roof details and it provides feature information that is used to optimise the traversal of the ground scan.

Despite this the greatest limitation of CUBE's use of MAMMAL is the 2.5D parallax nature of the result which means that underpasses such as tunnels and bridges cannot be fully accounted for using the airborne scan alone. This is also prob-

lematic for balconies and similar overhanging features that occlude features beneath them and manifests as unstructured facade edge points that deviate from the airborne mass representation. As such the primary improvement (necessary to produce a more robust operator) is dealing with the parallax limitation. One strategy to would be considering the unstructured scan data and refactoring the airborne mass-models in order to take into account areas that were occluded due to the top-down perspective. This would involve considering the density based wall map (constructed during registration) and isolating *interior* walls in the ground scan (verticality filtered wall points that are enclosed by a building footprint but lack a corresponding edge in the airborne massing).

To revise, CUBE invokes MAMMAL in order to produce sparse 2.5D mass models from the input airborne scan data. These massing models form the basis of the reconstructed scene. Once this process is complete CUBE then proceeds to reconstruct the ground based scan data. This is covered next.

Facades: Reconstruct Ground Models

With airborne mass-model recovery complete, CUBE preceeds to recover ground-based surface-elements models to append to the facades of each building. This is a 3D facade-model reconstruction stage that is driven by the previously defined ARROW algorithm. It exists to order to recover semantically rich facade geometry from each cluster of points that lie within range of each mass-model's constituent walls. One key problem is that ARROW expects clusters of points corresponding to single facades, whilst the ground scan contains numerous facades, vegetation and clutter which must be removed before ARROW invocation. As such this stage is achieved through a precursory guided segmentation of the unstructured scan and then feeding each cluster of segmented facade points into ARROW.

The inputs to this stage are the unstructured ground-based scan and the reconstructed airborne mass models. The output is a set of semantized 3D facade descriptors - with each descriptor composed of set of surface elements.

The proposed process boils down to two intuitive steps. First for each mass-model CUBE *grabs* clusters of points that are within a user-supplied distance from its constituent walls. Then for each segmented cluster of facade points CUBE passes the cluster to ARROW and associates the resultant facade descriptor with the corresponding mass-model.

The pseudo code following outlines the implementation of this stage:

1: $ret \leftarrow \{\}$
2: **for** $each\ mass\_model\ (M)\ in\ scene\_graph$ **do**
3: **for** $each\ wall\ (w)\ in\ M$ **do**
4: $fp \leftarrow grab\_points\_XZ(w, grab\_radius, pts)$
5: **if** $dense\_cluster(fp)$ **then**
6: $facade \leftarrow ARROW(fp, arrow_{args})$
7: $ret \rightarrow add(facade)$
8: **end if**
9: **end for**
10: **end for**
11: **return** ret

The advantages of this strategy largely boil down to ARROW automatically recovering clean, compact semantically-rich facade models from the unstructured scan. This is in itself a challenging problem. However more over in terms of efficiency, this method offers several benefits over existing methods. Firstly it is selective about what it reconstructs. Secondly it is trivial to parallelise since each facade descriptor can be processed independently of the others. The only time they must be considered together is in the final unification step. For example when using a library such as OpenMP - one can simply add *#pragma omp parallel for* directives to the two nested for-loops (see pseudo-code) - to execute over multiple cores.

The limitation of CUBE's ground reconstruction strategy is that it requires reasonably high-quality input pointsets. Missing data can be detrimental and generally results in missing surface-elements. Essentially because this is data-driven it does not make-up or synthesise geometry where there is none in the input. As such if a region of a facade is omited in the scan, then any windows and doors in the region will be omitted in the output model. Another limitation (related to this) is that low resolution scans (captured from too great a distance) may not provide enough detail for the stable extraction of connected components and surface-element split-logics. Since ARROW relies largely on depth variance, it is important that the ground-scan captures such. However the flip-side of this is that the technique for segmenting ground scans using the airborne massing models (dubbed DRAPES - DEM Reconstruction for Accelerated Partitioning of Engineering Surfaces) is stable in the presence of missing data on a particular facade. Meaning the surface points can still be isolated even with partial data.

To revise, CUBE's ground reconstruction stage is designed to produce compact facade detail models from the unstructured ground scan by efficiently segmenting the data using the airborne massing models and feeding individual facades into ARROW. Given that each complete exterior model is characterised as mass and surface - the final stage of CUBE is to unify the two types of geometry in order to produce a single merged model for each building.

Unify: Airborne and Ground Models

The last stage of CUBE is to merge the airborne massing models with the ground based facade models to produce complete building shell models.

The shell-models are the result of appending the surface elements from the previously reconstructed facade-models to their corresponding walls in the airborne mass-models and re-invoking ARROW's *clip* routine to revise each massing model's walls with the appropriate apertures. The process is applied iteratively for each facade model recovered by the previous stage.

The input to unification is a set of mass-models (and associated descriptors) and a set (of sets) of facade-models (and associated descriptors). The output is a set of building shell models - each represented as a polygon mesh.

This represents the simplest stage of CUBE and is responsible for composing the two types of architectural model recovered by MAMMAL and ARROW in order to yield complete building exteriors. This is essentially an associative task coupled with a re-clip. It involves *snapping* each surface-element to its guiding wall in its airborne mass model and then replacing each single face facade wall in the mass model with apperture clipped versions.

One positive feature of this is that it degrades gracefully, in the sense that the worst case outcome for a building is the airborne massing model. This can occur as a result of missing data in the unstructured ground scan.

The underlying idea in CUBE's unification is that by maintaining a high-level (and hierarchical) representation of the reconstructed data the process of appending the windows and doors to walls is simpler to coordinate than when applied to unsemantized 3D surface reconstructions. This effectively enables CUBE to *copy-and-paste* the surface-detail models onto associated mass models directly. Meaning there is no need for surface to surface registration [12] or additional watertighting methods since merging is handled at the level of objects rather than primitives. Another desirable aspect is the re-use of ARROW's aperture-clipper. Since all the logic of producing high-quality cut-out models is already embedded in ARROW, no additional routines are required to coordinate the mass-model refactoring that creates window and door apertures for the airborne models.

Despite this the biggest limitation is that the parallax nature of the massing-models means certain types of facade present in the ground scan (those that are occluded aerially) will be omitted from the result. Currently CUBE uses un-refactored massing models from MAMMAL as the basis of the scene graph. A crucial improvement to this would be to consider the density-feature-map used during registration post airborne reconstruction in order to determine (first) facades that are not represented by the mass models and (secondly) inject appropriate wall representations. This would have a two-fold effect. Firstly it would combat parallax occlusion - secondly it acts as a fall-back for temporally varying datasets, for which facades present in the ground scan may be missing in the airborne scan.

Summary

This section outlined an intuitive approach to multi-modal merging of airborne and ground building models derived automatically from laser-scans. It considered the practical requirements of each stage (registration, airborne reconstruction, ground reconstruction and merging) and detailed the potential sources of error and the fundamental limitations of the acceleration strategies employed. It considered extensions that seek to improve CUBE's performance - both in terms of completeness and robustness to lower-quality input point-sets. This chapter also discussed the inherent issues in attempting to merge temporally varying airborne and ground representations - from which one can deduce that minimising the temporal difference (delay) between the acquisition of airborne and ground representations is useful in limiting potential sources of ambiguity in subsequent reconstruction tasks.

Note: vitally that CUBE is still under active development - and as such these early-stage discussions do not necessarily represent the polished results sought by this research but rather the state of work in progress.

5.1.4 Preliminary Results: Tower-Bridge

Early stage results from CUBE's development are outlined next - and are supplemented by the high-level findings of this investigative track to date.

The dataset used to develop and debug CUBE represents the area surrounding Tower-Bridge in London. It contains structured airborne DEM points (at 50cm/4ppm resolution) acquired from the Environments Agency [47] and an unstructured com-

posite ground scan (registered from three seperate scans that have Tower-Bridge as the common focal point) - with RGB-D colour information - provided by this project's industrial sponsor. The ground scan exhibits partial building point clusters of variable density - however critically there are few roof points present.

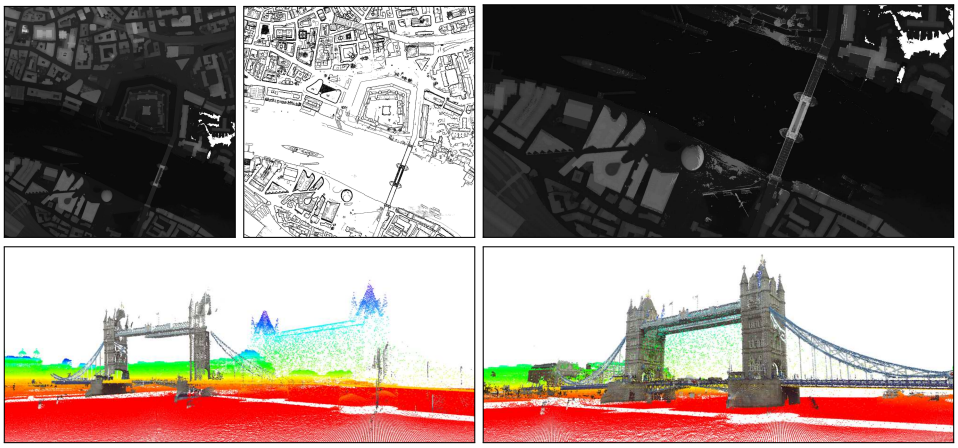

Figure 5.3: *registration of the structured airborne and unstructured ground laser-scans of the Tower-Bridge area employed in CUBE's development - illustrating (left) the unaligned inputs and (right) the outcome of CUBE's feature-based alignment*

Figure 5.3 illustrates the outcome of CUBE's registration whilst figure 5.4 illustrates the result of DRAPES (the guided facade segmentation technique).

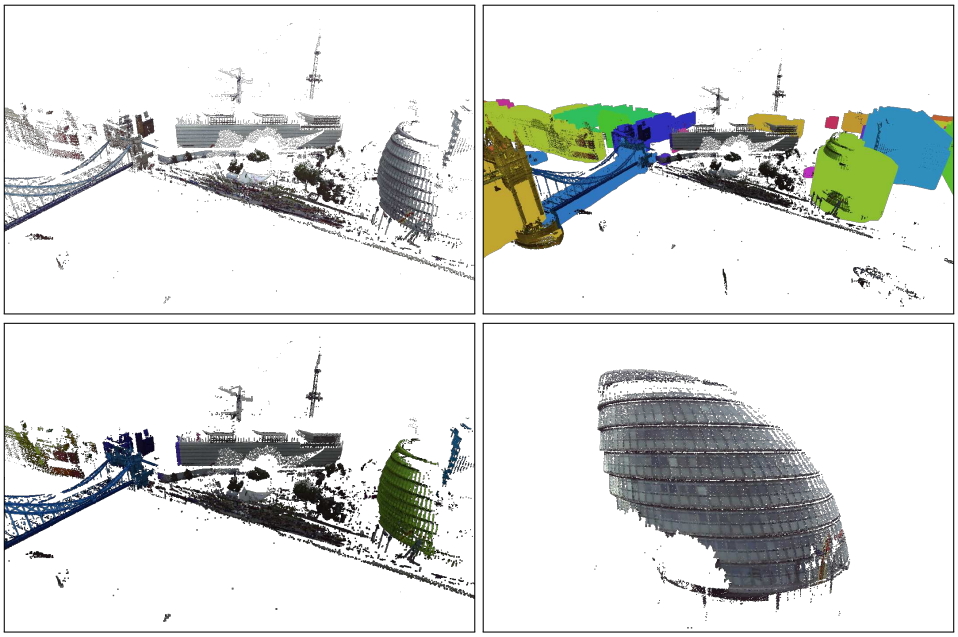

Figure 5.4: *automatic segmentation of unstructured ground scan guided by airborne massing models (DRAPES) - illustrating (from top-left to bottom-right) the input ground-scan points (with RGB scan colours), the ground scan alongside airborne mass model volumes (coloured by building mass), the ground scan points (coloured by mass-group - the result of DRAPES' point-grabbing operation), and (finally) a close up of a challenging curved partial facade cluster that is automatically segmented from the ground-scan by this technique.*

These early stage results demonstrate the validity of the first two and half stages in CUBE. However they also reveal the limitations of the current implementation. In particular the fact that only facades that have corresponding masses are isolated - (see figure 5.4) whenever there is temporal disparity between the scans.

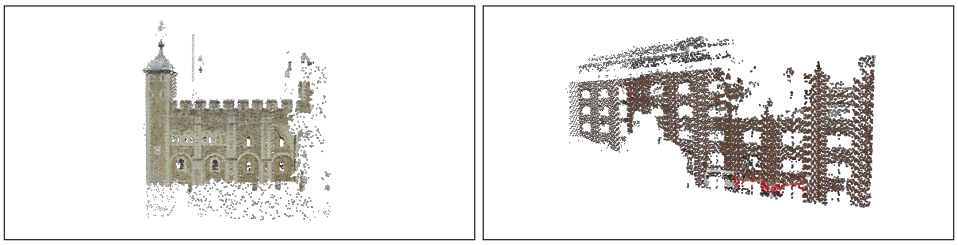

Figure 5.5: *further facades automatically extracted by DRAPES for the Tower-Bridge dataset - note in particular the lower density and regions of missing data in these representations relative to the Roslyn-Mews dataset employed in the previous chapter*

There are three main observations drawn from the development of CUBE to date.

- DRAPES: yields strong facade segmentation results for both dense and sparse building point clusters in the presence of planarity and curvature. This component of CUBE responds surprisingly well to arbitrary facades in the Tower-Bridge dataset irrespective of density and sampling artefacts (figures 5.4 and 5.5). However DRAPES' ability to isolate more than regular planar facades leads naturally to the requirement for CUBE's ground reconstruction stage to be able to polygonise such elements. In particular this means that the (currently plane-based) SDF-split employed by ARROW must be augmented to take into account non-linear facades via a projective unwrapping routine.

- Temporal Disparity: between airborne and ground introduces ambiguity in the problem specification and hinders detection and recall of elements in the ground scan. In this sense the completeness of the scene representations extracted by CUBE relies on there being a strict temporal correspondence between the airborne and ground scans. What this means is that the accelerations applied will only work robustly for airborne and ground scans captured at roughly the same time. The greater the delay period between their acquisition the greater the likeihood of there being disparity between them which interferes with CUBE's ability to efficiently produce a complete model.

- Surface-Element Recovery: represents the key bottleneck to large-scale multi-modal reconstruction because as the scope of a sampled region increases (typically) the level of detail captured by a scan for each surface-element decreases. Essentially the greater the distance from a facade the scanner is the fewer points correspond to window and door frames - which means less depth information is available to segment and model each element.

5.1.5 Discussion and Summary

This section outlined CUBE (that is: Complete Unified Building Exteriors) - a practical technique for efficient automatic building reconstruction from multi-modal laser-scans. The underlying idea in CUBE is to exploit MAMMAL in concert with ARROW in order to address the two key sub-problems - reconstructing building masses (roofs) and reconstructing building surfaces (facades).

By guiding processing of the unstructured ground scan with knowledge derived from the reconstructed airborne scan CUBE aims to minimise the complexity of the task. However this increased efficiency comes at the potential cost of the completeness of a scene's descriptor relative to an exhaustive traversal of the ground scan. In particular this means that temporal disparity between the two representations can result in facades that are not present in the ground scan being omitted from reconstruction. Logically the simplest means to combat this is to use airborne and ground scans of a region for which the delay in their acquisition is minimal. However in practise this may not always be possible - especially in the case of scans acquired from open-access repositories. As such alongside dealing with the parallax-occlusion problem (refactoring 2.5D masses to take into account underpasses and overhanging features such as balconies), a key requirement for CUBE's further development is an efficient non-stochastic wall-extraction operator for unstructured points that responds well both to planarity and curvature.

Critically - although this final unifying component of the proposed automatic temporal update scheme (sought by this research) demonstrates some potential - there remain many answered questions and unaddressed problems that currently limit its overall utility. These stand as areas for continued investigation.

Finally the key ideas underlying CUBE are revised below for reference:

- **Feature-Based Registration of Airborne and Ground Laser-Scans** - to combat the multi-modal weak-correspondence nature of the input scans.
- **DEM Reconstruction for Accelerated Partitioning of Engineering Surfaces** (DRAPES) - an intuitive facade segmentation method that guides the partitioning of facades with airborne massing models.
- **Hierarchical Semantized IR to enable 'copy-and-paste' Unification** - based on preserving an object level representation of the structure of each facade (a set of surface-element descriptors) and associative re-clipping of walls.
- **Re-Cycling of MAMMAL, MARS, QUALM, GRAILS, ARROW, SLADE**

5.2 Automatic Temporal Updates

and all together...

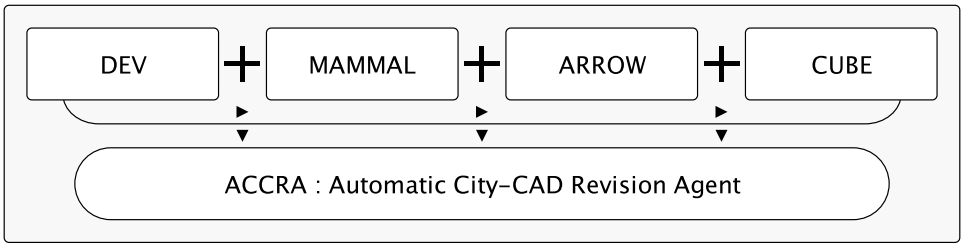

Figure 5.6: *overview of the key components in automatic temporal updates*

5.2.1 Overview

This section discusses how to chain together the four key techniques for semantic change detection, airborne reconstruction, ground reconstruction and multi-modal merging, to form a stand-alone artificially intelligent CAD technician, that not only understands how to create sparse geometry from point-cloud data but further embodies the logic necessary to be selective about that which it reconstructs. In essence the *Automatic City-CAD Reconstruction Agent* is a minimalist, whose awareness of the nature of urban scenes enables it to apply *lazy-evaluation* strategies to the task of automatic content generation and maintenance.

At a high level, one can think of ACCRA (Automatic-City-CAD-Revision-Agent) as a script that automatically chains the inputs and outputs of each of the previously defined algorithmic components. This temporal update pipeline enables continuous architectural CAD revision by analysing changes between an out-dated CAD model and newly acquired point-clouds (using DEV), and creating new building geometry for detected extensions, constructions and replacements (using MAMMAL, ARROW and CUBE). Figure 5.6 illustrates these key components.

To help clarify examples of concrete use-cases for the automatic temporal revision agent described in this section - one could conceive using it:

1. As a data-driven automatic urban CAD model updater - which is the primary use-case of ACCRA and the driving aim of this research.
2. As part of a geometric version-control system for temporal 3D city-models - within which the present-day model is maintained alongside a per-building change-log that enables a technician to step through instances of the model at various points in time. This supports analysis of the development (evolution) of an urban region over time.

The rest of this section specifies the main requirements of the solution, the methodology underlying its operation and discusses the current state of development.

5.2.2 Key Requirements

For reference the key objectives, requirements and desired behavioural characteristics of the outlined reconstruction agent are outlined following.

- Geometric Accuracy : such that revisions made to the out-dated CAD model accurately reflect the geometry represented by the up-to-date point-clouds. This means error-bounded reconstruction results.
- Computational Efficiency : in particular the desire for linear growth in runtime as a product of the size of the inputs and low-memory-use.
- Scalability : in terms of the ability to process both large airborne regions 100km^2+ and large unstructured scans 100,000,000pts+ whilst maintaining quasi-linear performance. This means removing latency such that additional (coordinatory) processing time isn't introduced as the size of the input representations grows extremely large.
- Tractable : change-detection results and corresponding revisions. In other words ACCRA should not obfuscate the process of finding modelling errors or creating updated building geometry. A technician should be able to trivially map the analytic outcome of change-detection to the alterations made to the out-dated CAD model.
- Robust : to both high and low frequency sensing noise, low model quality, partial and missing data and geometric degeneracy.
- Support for Parallel Processing : such that the processes of airborne mass reconstruction and ground facade reconstruction may be coordinated across multiple cores. As a further requirement - suitability to execution on the GPU and using distributed architectures.

The steps necessary to address these requirements are outlined next.

5.2.3 Methodology

The key stages in ACCRA can be broken down into the following five steps:

1. Load, Pre-Process and Register Input CAD and Scan Data
2. Isolate and Classify Significant Temporal Changes
3. For Each Temporal Change - Reconstruct Associated Point Clusters
4. For Each Temporal Change - Integrate Reconstructed Models
5. Validate the Updated CAD Model and Return

Another way to rationalise the key stages in ACCRA is in terms of the three-step strategy: analyse, reconstruct, compose - outlined following:

1. Analyse: the input data to isolate and classify temporal errors.
2. Reconstruct: up-to-date geometry for each identified error.
3. Compose: the CAD model with the the reconstructed models.

Both of these decompositions describe the intended operation of ACCRA abstractly. The first maps loosely to the logical algorithmic flow of ACCRA, whilst the second denotes a generalised ideological break-down.

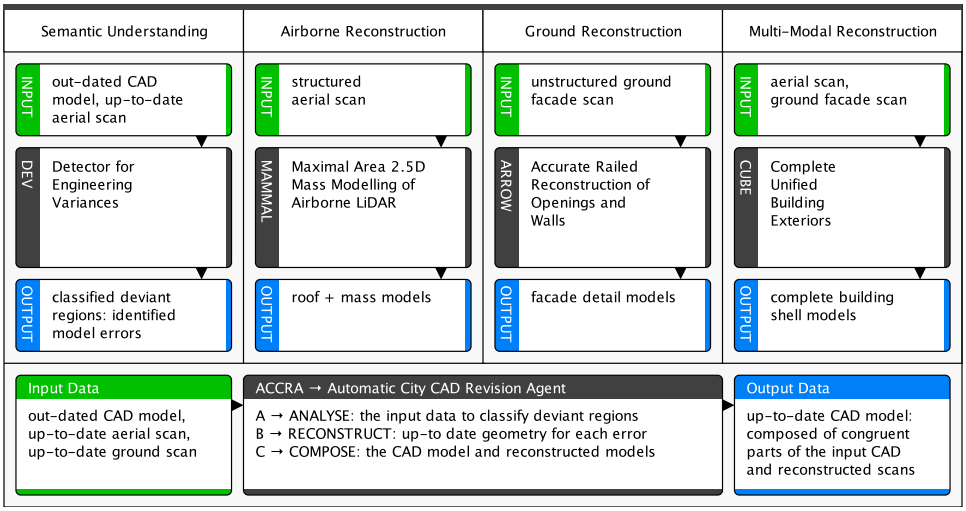

Figure 5.7: *structure of the automatic city-cad revision agent (ACCRA) - in terms of its inputs (green), processing stages (dark) and outputs (blue) - for each of its constituent sub-components (top) and as a stand-alone reconstructive unit (bottom).*

To help further clarify the behaviour of ACCRA figure 5.7 outlines its structure.

5.2.4 Discussion

Foremost the performance of ACCRA is largely controlled by the efficacy of each of its constituent components (DEV, MAMMAL, ARROW and CUBE). For example the application of selective reconstruction and the ability to efficiently track modelling errors is contingent on the performance of DEV. Whilst the quality and compactification of the up-dated models rests upon the shoulders of MAMMAL, ARROW and CUBE. Although in principle MAMMAL and ARROW can be used in isolation, CUBE depends on both of them and as such efficient complete building (mass+surface) revision is only feasible as a product of their joint exploitation.

In this regard the key point is that whilst each operator works reasonably well in isolation, the process of amalgamating them under the banner of an efficient intelligent revision agent remains a problem under active investigation.

Further regarding the present state of ACCRA, another key missing piece is a suitable testing dataset. As stated ACCRA calls for three types of input: an out-dated CAD model, an up-to-date airborne laser-scan and and an up-to-date ground-based laser-scan. Although synthetic data-sets could be used, in principle one requires several real-world examples in order to accurately quantify the performance of the approach. This means that a key pre-requisite to meaningful academic evaluation is acquiring suitable multi-modal input data.

Nonetheless - the ideas and observations underlying ACCRA's development should be quite clear. The core aim is to automatically maintain high-quality 3D building models of rapidly changing urban environments and the operators presented correspond to the principal sub-components of a potentially viable solution.

5.3 The Remaining Open Problems

This section discusses the remaining open problems that pervade each of the operators proposed by this research. It aims to provide a synopsis of the key aspects of each component in ACCRA that requires further investigation.

5.3.1 Semantic Change Detection

In relation to semantic change detection - the key issue with the current approach is that the change detector operates at the level of buildings rather than components. As discussed in chapter two, the utility of the change detector could be significantly enhanced if deviant regions were decomposed into individual variances prior to classification. The reason this remains an open problem is that determining a decomposition strategy is a non-trivial task given that it requires multiple variance classifications to be made for each building which in turn increases the complexity of any subsequent revision process. Additionally it means that the evaluation order of each deviant component of a building becomes critical. One potential way to address this would be to sort each deviance by its scope or significance. Nonetheless this remains an unsolved problem requiring further experimentation.

The other key remaining problem in the change detector is determining how to automatically alter the weights associated with the thresholds to the boolean predicates that distinguish each class of variance. In particular, chapter two identified the need for the change-detector to relax the tolerance for variance as a product of the size (extents) of a building. Now although the idea is quite simple - in reality the implementation is less trivial - and would require a linear or non-linear function derived empirically through the consideration of a representative set of old and new urban representations. In other words, determining the relationship between the size of a building and the classifiers tolerance for deviations cannot be addressed solely as a product of the input - and may mandate off-line strategies such as (but not strictly limited to) machine learning using NNs or convolutional NNs.

Further the mutually exclusive nature of the change detector also has implicit limitations - in particular chapter two discussed some of the problems of forcing the classification of the *position* operator by increasing the tolerance for variance between the out-dated and newly-acquired representations - and noted that such acts typically decrease the overall response of the classifier's other deviance classes relative to human-labelled variance data.

5.3.2 Airborne Reconstruction

In terms of the airborne reconstruction - two key on-going problems underlie the performance of the MAMMAL algorithm. These are segmentation and vectorisation. Given that the results of vectorisation are largely a product of the segmentation process - the key area that requires further refinement is the automatic segmentation of airborne point-clouds. Despite this - the vectorisation stage (considered in isolation) could still be improved in order to produce higher-quality 2D vector shape arrangements. The critical concern with both stages (and the reason they remain open problems) is the fact that neither is amenable to a closed-form analytic solution. Although the segmentation and vectorisation performance of the

current implementation yields reasonably satisfactory results - identifying methods of further mitigating the effects of sensing noise remains a high priority.

5.3.3 Ground Reconstruction

To dominant problem with the automatic facade reconstruction is the stable resolution of each surface-element's interior split-axes. The SLADE algorithm is the component responsible for addressing this problem, however its performance on irregular sash divisions is still unsatisfactory. Ultimately the difficulty lies in the fact that there is not always a direct relationship between a window's segmented point-cluster and the desired 2D division of space. Generally researchers have combatted such issues with the use of template fitting, priors and heuristic constraints. Now although chapter four discussed examples of such methods and their applicability to the overall arrangement of a facade - there remains a lack of a definitive solution to the problem at the level of individual surface-elements. Essentially - in the case of sashes composed of rectilinear edges the response of SLADE is reasonable, however mining higher-order features such as circular arcs and bezier curves without the use of stochastic sampling is still to be addressed.

The other key unsolved problem is refactoring the ARROW algorithm such that it can deal with lower-quality input scans. The critical point is that ARROW relies upon depth variance in order to facilitate stable data-driven segmentation - which means it requires high-fidelity input scans. However (as discussed in chapters four and five) this requirement will not always be possible especially in the case of facade scans surveyed from large distances. As such it would be desirable for ARROW to possess an alternative (fall-back) strategy for isolating windows and doors even in the presence of lower density scan-data. The problem is that the identification of such a strategy has (up to this point) proven itself elusive.

5.3.4 Multi-Modal Merging

Two key problems remain unsolved in multi-modal reconstruction. The first is identifying and addressing temporal differences that may be present between the airborne and ground laser-scan due to the fact that they may have been surveyed at slightly different points in time. The second is automatically refactoring the reconstructed airborne models in order to account for non-parallax features that are not represented in the aerial scan but are present in the ground scan. The first problem is a product of the ambiguity inherent to distinguishing temporal differences from partial and missing data. The second problem results from the invalidation of the parallax (top-down) assumption that is employed to constrain and enhance the efficiency of the reconstruction process. Now the reason the first problem is tricky to address is because (unlike semantic change-detection) the correspondence between the airborne and ground scans is weak - due to the multi-modal nature of the inputs. Hence one must account for the fact that the airborne scan may be missing parts present in the ground scan, and the ground-scan may be missing parts present in the airborne scan - without a definitive means of determining which type of scan is correct. Now the key challenge presented by the second problem is largely a product of the increase in processing requirements necessary to transform the top-down massing into a full-blown 3D mass. This not only calls for a more generalised (and hence expensive) error-measure, but it also

demands a free-form abstract representation of a building which in turn limits the parametric extensions and revisions that can be efficiently made - requiring instead traditional polygon-mesh manipulations. Essentially - whilst it is feasible to cut-out non-parallax features from the airborne massing models - guided by the ground-scan data - in practice doing so whilst maintaining the performance characteristics of the current methods is a non-trivial task.

5.3.5 Summary of Unresolved Problems

In summary the following list revises the vital remaining problems discussed in this section that require addressing through continued investigation.

- Scale-Dependant Variance Tolerances for Change-Detection.
- Identifying Multiple Modes of Change per Building.
- Enhancing Airborne Segmentation and Vectorisation.
- Recovery of Sash Divisions for Irregular Surface-Elements.
- Facade Segmentation and Reconstruction from Lower Quality Scans.
- Distinguishing Temporal Multi-Modal Variances from Partial-Data.
- Efficient Airborne Mass Refactoring for Non-Parallax Features.

5.4 Further Investigations

The final part of this chapter discusses some of the auxiliary investigations related to (but distinct from) the task of automatic building reconstruction and temporal maintenance using point-clouds. These areas relate the to production of interactive simulations of the physical world - and in particular the recovery of the supplementary assets that enrich visualisations.

For each of these future investigative topics a synopsis of the core aims and constituent problems is provided in the form of a high-level abstract.

Recovering Material-Shaders

This investigative track seeks to recover procedural material shaders from RGB-D scans in order to enhance the rendering of building geometry.

Note: that many existing operators (such as [12]) already support the recovery of textured models using the colour data from photographs or laser-scans. However the critical problem with these approaches is that they require high-density texture-representations in order to accurately reflect the shading of an object. The key idea of this investigation is to transform automatically recovered texture-mapped materials into procedurally defined materials - which are similar in vein to Alegorithmic's *substances* - in the sense that they define a surfaces material properties functionally as opposed to explicitly with raster images.

The benefit of this strategy is dramatically reducing the memory requirements for the storage and rendering of reconstructed building models. This is predicated on the underlying insight that - at city-scale, texture memory represents the critical bottleneck for real-time rendering - yet by characterising material properties procedurally this issue can be largely resolved.

At a high-level there are two basic strategies for addressing this problem and they can be loosely thought of as being data-driven or model-driven.

The data-driven approach involves analysing clusters of XYZ-RGB points (where each cluster is assumed to be belonging to a single un-obscured region of a surface in the scene) and automatically constructing a unique shader for each distinct material identified in a scene. This is quite a challenging problem given that it effectively requires one to teach the computer how to write shaders in very much a *by-example* manner. The model-driven (library-based) strategy (on the other hand) is similar to template fitting, in the sense that one begins with a set of parameterised shaders for the prominent types of material in architectural scenes (wood, glass, stone, brick, metal, plastic and so on) and frames the task as identifying the class of, and inputs parameters for, each material in a scene.

The critical challenge inherent to both approaches (and likely the reason that researchers have avoided this problem) is that one cannot rely upon a geometric similarity measure in order to determine the aesthetic error between the input RGB data and the output material shader. This problem not only calls for BDRF analysis - it is also subject to the limitation that the resulting shader would be less visually accurate than a texture-map pasted directly onto a surface. However from the perspective of visual quality, this is not a critical problem, because a procedural material is implicitly resolution independent which tends to result in higher fidelity

close-up visualisations. Further more, by exploiting RGB-D scans to mine material properties as opposed to images a greater amount of spatial information is present which has the potential to enhance the recovery of semi-regular structures and patterns (such as brick and stone-work).

Reconstructing Street-Furniture

This area for further investigation aims to recover 3D street-furniture models of objects such as lamp-posts, park-benches, monuments and statues from unstructured ground scans - in order to enrich the physical realism of scenes.

Note that whilst street-furniture models are typically not critical for the simulation of viewing corridors or spatial analysis - they play a major role in the production of photo-real still images and animated sequences.

The critical challenge is that although the key types of object may be known ahead-of-time, the geometric form of each instance can vary dramatically across an urban region, meaning that template-driven strategies are less amenable to this problem. In order to produce accurate street furniture models, data-driven reconstruction strategies are required. Furthermore unlike buildings - which can be easily distinguished from clutter and vegetation based on spatial scale and their regularity - lamp-posts and similar objects as less easily distinguishable - mandating more involved filtering and fault-tolerance strategies.

Reconstructing Building-Interiors

The recovery of building interior geometry is predicated on the desire to be able to simulate and render interior architectural scenes. This research has focussed on the recovery of exterior architectural geometry - however building-shell models alone are insufficient in simulating virtual environments for which an end user is required to be able to navigate the rooms inside of a building.

At a high level one could decompose this problem into the recovery of three abstract types. The reconstruction of (i) internal structural and storage vertical features (such as walls, doors and shelving units), (ii) the reconstruction of floor-aligned features (such as tables, desks, cabinets, wardrobes, bedding and chairs) and (iii) the reconstruction of overhanging ceiling-aligned features (such as light-fixtures, air-conditioning units, smoke-detectors and cooker hoods). By treating the task in terms of these three object-classes (verticals, floor-aligned and ceiling-aligned) the process of instantiating geometry for a building interior (given unstructured point-cloud data) becomes amenable to constraint-based shape arrangement. However the critical challenge (that would still remain) is dealing with the clutter and disorganisation that often pervades building interiors. Furthermore - whilst such an ontology would be useful for dealing with individual rooms, there would still be the problem of coordinating the recovery of auxiliary features such as stairways, complex entrances (such as revolving doors) and elevators.

The underlying geometric problems that make sparse interior reconstruction such a juicy academic problem are dealing with occlusions and irregularity. However it is worth noting that knowledge regarding a building's exterior structure has the potential to aid such processes - for example by resolving ambiguities that could arise as a product of features such as duplexes.

Chapter 6

Conclusions

Finally this chapter concludes this thesis. It summarises the technical and theoretical contributions, the key *take-home* insights and acknowledges the parties that supported this research. It includes two peer-reviewed research posters presented at SIGGRAPH 2016, and is followed by the bibliography.

6.1 Contributions

This section revises the key contributions to knowledge proposed by this research. They address key problems in the domain of architectural reconstruction from laser-scanned point-clouds. These are semantic-change-detection, airborne mass reconstruction and ground facade reconstruction. They are divided into two categories - algorithmic and ideological. The algorithmic contributions are arranged according to the problem they address - followed by the ideological contributions.

6.1.1 Algorithms

The following represent the key technical developments resulting from this research. They are self contained algorithms for addressing problems such as automatic segmentation, boundary-extraction and model-optimisation within architectural reconstruction from laser-scanned point-clouds.

Semantic Change Detection

- **DEV** - this thesis introduced DEV (Detector for Engineering Variance) - a novel classifier for semantic change detection between multi-modal architectural datasets. DEV provides a closed-form analytic method of identifying instances of *new-constructions*, *demolitions (removals)*, *extensions*, *reductions*, *re-positions* and *replacements* in out-dated CAD models given up-to-date airborne laser-scans. The performance of DEV as an error-detector has been evaluated using the City of Bath CAD and LiDAR dataset and the results demonstrate stable automatic temporal error identification. The critical benefit over existing architectural change detectors is that DEV can associate point-level deviances with high-level *actionable* corrections. This enables an optimisation to data-driven city modelling termed *selective-reconstruction*.

Airborne Reconstruction

- **MAMMAL** - this thesis introduced MAMMAL (Maximal Area 2.5D Mass Modelling of Airborne LiDAR) - an algorithm for fast, accurate and sparse automatic city-reconstruction from digital elevation models. MAMMAL efficiently *segments*, *vectorises*, *projects* and *optimises* data-driven massing models using the *maximal-area : minimum-primitives* principle. The vital benefit over the dominant 2.5D methods, is that MAMMAL's optimisation automatically produces parametric building geometry that can be manipulated interactively. MAMMAL's performance has been analysed using airborne scans of the cities of Manchester (at 25cm point-spacing), London (at 50cm point-spacing) and Bath (at 1m point-spacing). The results demonstrate robust recovery of compact parallax massing models that capture roof structures and details at various spatial scales, in quasi-linear time. This represents a significant speed-up in the time taken to produce 3D building assets from airborne scans. The true power of MAMMAL results from the amalgamation of data-driven projections with dynamic model-driven procedural geometry.

- **MARS** - this thesis introduced MARS (Maximal Area Roof-Shape Segmentation) - a low-level image-based segmentation operator for identifying building roof-shapes in aerial scans, based on the difference of elevation models and scale-driven explicit suppression of undesirable roof-features that would otherwise bloat the result and introduce latency in polygonisation. MARS combines local connectivity measures with a generalised graph region traversal in order to efficiently isolate the salient architectural features of airborne scans whilst directly supporting the *maximal-area : minimum primitives* principle. The key advantage of MARS (relative to pre-existing maximal-area segmentation methods) is its significantly reduced execution time.

- **GROVE** - this thesis introduced GROVE (Graph-Refinement Operator for Vector Extraction) - a raster-to-vector converter that preserves the topology of neighbouring roof-shapes by detecting *critical-points* which are anchors used to constrain the boundary extraction process and ensure watertight shape-nets are constructed. Unlike the prevalent architectural edge-detectors, GROVE does not stochastically fit lines. Rather each shape-net is a repeatable product of the input raster segmentation and user-supplied error tolerance. GROVE automatically produces vector shapes for 2D mapping with greater architectural detail than is present in off-the-shelf maps such as OSM. In this sense the output of GROVE is closer to a 2D master-plan.

- **GRAILS** - this thesis introduced GRAILS (Graph-Railed Approximate Interior Linear Spine) - a geometric feature-detector that extracts sparse medial shape spines from point-sets for the purpose of automatic terrace modelling. Each GRAIL spine is the product of computing the maximal length non-cyclical path of an interior graph of medial axis convergence points. GRAILS produces higher quality vectors from low resolution scans than data-driven or stochastic line-fitting strategies. In particular it can handle non-monotonicity and approximate curvature robustly using piecewise linear arrangements.

- **QUALM** - this thesis introduced QUALM (Quick Unconstrained Approximate L-Shape Method) - a domain-specific boundary-extraction and polygon simplification function for efficiently extracting common rectilinear architectural

footprints from aerial scans. QUALM produces sparse L, T and S shaped building boundaries (with a guarantee on the maximum number of vertices: ≤ 12). Its primary purpose is to support automatic 2D and 3D map updating. Despite its simple heuristic nature QUALM is surprisingly robust at various resolutions, producing master-plan quality building boundaries in linear-time.

Ground Reconstruction

- **ARROW** - this thesis introduced ARROW (Accurate Railed Reconstruction of Openings and Walls) - a data-driven algorithm for fast, accurate and sparse automatic 3D facade model reconstruction from unstructured ground laser-scans. ARROW produces compact semantized facade models which are suitable for physical simulation and building-energy analysis. The power of ARROW is a result of marrying the accuracy and efficiency of a data-driven method with the topological quality of a model-driven method. ARROW exploits a novel procedural representation - the surface-element in order to efficiently recover both regular and irregular 3D window and door models.
- **SLADE** - this thesis introduced SLADE (Split-Logic Axes Detector and Extractor) - an algorithm for transforming clusters of points corresponding to a window or door into 2D polygonal split-structure shapes. SLADE is the key component in ensuring the structural accuracy of each window and door model generated by the ARROW algorithm.

Alongside these procedural operators, this research introduced and exploited a number of high-level ideological concepts which are summarised next.

6.1.2 Theoretical Abstractions, Principles and Paradigms

To facilitate fast, accurate and sparse architectural model reconstruction from airborne and ground laser-scans, this research exploited a number of insights and axiomatic principles regarding the nature of architectural representations in different modalities. These observations and abstractions enhanced the efficiency and efficacy of the methods presented.

Boolean Logic for Semantic Change Detection

The thesis proposed the use of boolean logic for detecting semantic errors in city-CAD datasets, that (unlike pre-existing change detectors) operates at the level of objects (rather than points). Boolean logic is not only a fundamental component of geometric algebra, but further largely negates the need for data-dependant training strategies. This closed form approach is distinct from alternative methods due to its purely geometric nature. Further enhancements to this strategy include scale-based adaptivity and the detection of multiple modes of change per building.

Maximal-Area, Minimum-Primitives : Under an Error Tolerance

This thesis exploited a simple paradigm for 2.5D mass modelling that seeks to formalise the key aspect of manually constructed building models. The maximal-area minimum-primitives principle is analogous to the *sparsest-under-an-error-tolerance* and captures the notion that human CAD technicians seek to represent

geometric form minimalistically. Compared to Zhou et al's principle regarding the presence of sharp edges characterising architecture - this principle is far more generalised (as it also handles curvature) and as a result is also applicable to other reconstructive tasks. The critical point is that there is no constraint placed on the types of features represented - rather the principle addresses the manner in which an algorithm uses geometric primitives - sparingly.

Depth-Buffer Error-Measure for Parallax Displacement Fields

This thesis employed a fast parallax error-measure for non-linear optimisation of 2.5D mass-models based on object-order rasterisation. This error-function is orders of magnitude faster than the prevalent point-to-plane and ray-casting equivalents - which enables the procedural optimisation kernel employed in airborne reconstruction. In essence, without this, efficient traversal of dynamic functional mass descriptors would be infeasible. The insight to draw from this is that: fast geometric-error measures facilitate data-driven fitting of non-linear generative modelling functions. In other words - by reducing the time it takes to measure error - one no longer has to guess randomly to efficiently traverse the solution space.

The Surface-Element

This thesis introduced a novel procedural primitive for the definition and automatic construction of 3D window and door models. The surface element is a data-driven function that formalises the notion of an aperture model using 2D vector-shapes and generalised polygonal split-logics in order to define and generate 3D generalised cylinders (3D sweeps). The utility of the abstraction results from its suitability to forward-chaining and backward-chaining modelling tasks.

Selective (Lazy) Reconstruction : Reconstructive Culling

This thesis proposed an analytically driven reconstruction optimisation informally dubbed *selective reconstruction* - wherein only buildings that exhibit significant alterations require re-modelling. This is driven by the detection of modelling errors between an out-dated CAD model and a newly acquired laser-scan. In a similar manner to view-frustum culling in rendering algorithms, reconstructive culling reduces the computational requirements for continuous temporal update to city-scale architectural datasets - by ensuring that automatic algorithms need only spend compute time modelling buildings that exhibit significant temporal variations.

6.2 Research Impact

This section discusses the impact of this research - both in terms of the industrial and academic effect it has had to date - and the anticipated continued impact.

6.2.1 Industrial Advancements

In relation to the industrial benefit of this research to this project's industrial sponsor to date - Cityscape Digital have already employed key aspects of the airborne reconstruction work on live projects to serve the purpose of cost-effective geo-spatial asset production. For example QUALM and GRAILS have been used in vectorising large portions of the City of London for which Cityscape lacked coverage and the cost of purchasing off-the-shelf models was prohibitive. The accompanying short paper: **Efficiently Extraction Extrusions from Airborne Range Scans for Animation and Visualisation of Urban Environments** provides a fuller account of the outcomes of one such instance where additional components of MAMMAL were used to produce 3D assets that were subsequently used in animated sequences produced by the company for a proposed development. The powerfully advantageous aspect of this particular example is that the input data used to create the assets (structured airborne point-clouds) cost Cityscape nothing!

Yes - the simple ideas relating to the formulation of fast, accurate and sparse deterministic building reconstruction methods - presented in this thesis - are already delivering a measurable gain to Cityscape both in terms of time and money saved.

In this sense it is fair to say that this research has had a positive impact upon its industrial sponsor and served Cityscape's progression. It has imbued the company with the know-how necessary to create some of their raw materials (compact building massing models) quickly and scalably from freely accessible point-clouds.

Note: however though that not all aspects of this research have reached the state of being used in production. In particular the facade reconstruction is still undergoing development as is the multi-modal merging of airborne and ground.

6.2.2 Academic Publications

In terms of the concrete academic deliverables of this research to date - this subsection documents two peer-reviewed short articles derived from this research - which were presented at SIGGRAPH '16. One tends to vectorisation in airborne mass reconstruction whilst the other deals with ground facade reconstruction.

Fast, Accurate and Sparse Automatic Facade Reconstruction from Unstructured Ground Laser-Scans
K. Edum-Fotwe, P. Shepherd, M. Brown, D. Harper, R. Dinnis
SIGGRAPH 16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4371-8/16/07 | DOI: http://dx.doi.org/10.1145/2945078.2945123

QUALM: Quick, Unconstrained, Approximate L-Shape Method
K. Edum-Fotwe, P. Shepherd, M. Brown, D. Harper, R. Dinnis
SIGGRAPH 16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4371-8/16/07 | DOI: http://dx.doi.org/10.1145/2945078.2945163

6.2.3 Looking Ahead - Potential Applications

Although it may take some time for some of the ideological contributions prescribed by this research to take root academically - industrially there are many immediate practical gains inherent to the data-driven strategies developed. From the application of the vectorisation routines (in 2D-map updating) through to the use of the semantic change detector (in temporal analytic tasks) and the mass-parameterisation logic (for top-down building modelling) - the utility of this research is not constrained to architectural visualisation, and has the potential to support the energy, transportation and infrastructural industries. Indeed any industrial pursuit that relies upon accurate geo-spatial assets is within the scope of the beneficiaries of this work. However in relation to Architecture, Engineering and Construction (AEC) specifically - the anticipated applications of this research include:

- Use of meta-data provided by MAMMAL's parametrics to perform semantic queries on CAD assets (for example: find all buildings in a particular region that possess curved roof components).
- Use of MAMMAL-styled functional descriptors to reduce storage and transmission requirements for city-models (LOD-ing and model compression).
- Use of DEV's logic to visualise temporal variations at city-scale (time-lapse animations of the structural evolution of an urban region).
- Use of DEV's logic to enable temporal geo-spatial queries (for example: find all buildings in an area that have been extended in the last two-years).
- Use of facades generated by ARROW's logic to perform solar-gain studies on buildings (which relies upon the accurate data-driven aperture locations).
- Use of ARROW's output to perform energy-efficiency simulations on buildings (again as a product of the accuracy of the window and door locations).
- Use of SLADE's split-axis-logic for reconstructing component-based window and door models for historically significant buildings - within preservation and heritage (for example irregular and/or unique ecclesiastic openings).
- Use of CUBE's building shell's within physically based interactive simulations and experiences - particularly for training, documentation and asset-management, analysis, planning and emergency response activities.

These are just a handful of the potential applications within architecture, engineering and construction. There are also (alongside these) all the pre-existing uses for 3D building models found in geo-spatial science and digital-media production.

6.3 Key Insights

Finally the key *take-home* insights and conclusions are summarised.

- Geometric Logic over Stochastic Sampling - this is Computational Geometry, rather than Computer Vision. There is still the requirement to mitigate sensing-noise, however fundamentally the vital problem associated with photogrammetry (recovering depth) has already been addressed. By operating directly on actively-sensed point-clouds the key requirement is not to guess where objects are but rather to effectively stitch together a sparse subset of the input. Despite the fact random-sampling is exploited in most of the current operators - this research points towards the fact that by omitting stochastic processes - equivalent sparse geometric results can be produced in a fraction of the time. This naturally leads to the next related insight.
- Determinism as a means to enable control of geometric topology and ensure computational efficiency and algorithmic dependability and to ease both the processes of development and performance-analysis. This thesis advocates that within the domain of architectural reconstruction - principled determinism is always preferable to stochastic guess-work. This is because principles can be improved, implementations can be optimised, however random-strategies fundamentally cannot be trivially controlled. This is the critical difference between this research and pre-existing sparse techniques. Every calculation, every algorithmic decision, every construct and predicate is computed deterministically. The proposed approaches are governed by logic. The implementations may be far from perfect, but the sheer fact that the methods CAN be (and currently are being) enhanced demonstrates the versatility that is still possible deterministically. The author is of the strong opinion that any reliance on random-sampling for architectural reconstruction from laser-scans, is a step backwards. As a species we previously relied upon RANSAC based paradigms due to the seeming complexity of the task. However with the continual advances in hardware and software patterns, the time has come to alleviate ourselves from the burden of trying to enhance random strategies - and focus on intuiting the fundamental logics that can be exploited to address problems in architectural reconstruction. Beyond topological control - another key benefit is limiting the exponential growth in execution time that is commonly witnessed as a product of the size of (the number of points in) the input laser scan. Determinism brings us closer to quasi-linear techniques - which are becoming increasingly important as the scope of the applications of active-sensing grow. Granted they may not be as simple to devise, however their behavioural benefits drastically out-weigh the burden/effort/load/-cost of their development.
- Segmentation is the most critical stage (in terms of semantic correctness and representational quality) for an automatic architectural reconstruction operator. This is because it feeds down to all subsequent processes and effectively controls the arrangement of geometries in polygonisation. Excellent segmentations yield excellent models. Poor segmentations yield poor models. Although this may seem obvious when stated - it is not always immediately apparent. The key challenge is that it is an ill-defined problem. Further still, fundamentally one does not require a good segmentation to minimise geometric error. For example dense-surface reconstruction yields accurate ge-

ometries without even considering the input at a component level. Hence it is not sufficient to simply consider the error associated with each segmentation to quantify it's quality. Whilst the maximal-area, minimum-primitives principle (exploited for the airborne reconstruction) addresses this to an extent, there is still the requirement for better performing high-level segmentation strategies and quality measures. As such (alongside vectorisation) this stands as one of the key areas of continued investigation.

- Exploitation of Dynamic Procedural Primitives (Data-Driven Generative Modelling Functions) as a means to dramatically expand the expressive scope of Library and Model based sparse building reconstruction paradigms. The caveat being that this relies heavily upon fast error-functions.
- Automatic building modelling is not the same as surface-reconstruction. High-quality models generally embody semantic meaning, reconstructed surfaces generally do not. This calls for more novel generalised data-driven modelling strategies, rather than refactored instances of the already well understood approaches to implicit and explicit surface reconstruction. In other words, within architectural modelling for visualisation and simulation one should seek to limit the use of dense surface representations - opting rather for error-bounded sparse, semantized representations.
- Input scan-quality (in terms of completeness, noise and mis-alignments) represents the key limiting factor in data-driven algorithms.
- Implementation in Hardware → in order to minimise time (turn-around) between scanning and geometric model generation - by enabling sparse 3D preview in realtime (on-site) during scan acquisition process. This represents the ultimate aim underlying the development of fast, accurate and sparse data-driven methods. Currently, the process of producing architectural assets from laser-scans is dominated by the time it takes to acquire the data and the time taken to transform the points into high-quality 3D models. The idea here is that if the time taken to reconstruct the points is negligible (near real-time) it becomes feasible to run the reconstruction at the same time the data is acquired. Essentially this would enable a surveying technician to inspect the reconstructed geometry generated from a laser-scan whilst they are still on-site and (if necessary) to re-scan immediately in order to combat issues such as occlusion - which would have the effect of minimising the delay between the scanning process and the creation of digital CAD assets.
- Limit use of Constraining Domain-Specific Modelling Priors - such as common snapping angles, the manhattan assumption, ubiquitous planarity, fixed-form regularity → in favour of loose/fuzzy/flexible data-driven logics. This naturally leads to the last related insight.
- Requirement for Effective/Imaginative Data-Driven Reconstructive Methods

Closing Thoughts / Concluding Remarks

The key aim of this research has been enabling automatic temporal updates to city-scale architectural models driven by fast, accurate and sparse building model reconstruction from airborne and ground laser-scans. In this regard, the developments presented move us closer to this aim. However, much still remains to be addressed. These final statements consider the vital open problems that must be resolved and the author's views on the future direction of this research.

The greatest continued challenge is extending the ARROW algorithm to handle partial and low-density ground scans. At present ARROW struggles to deal with scans captured from large distances and its integral sub-component (SLADE) could use significant improvement in terms of recovering irregular surface-elements. Realistically the time frame for bringing the performance of ARROW up to a professional standard may be several years. Nonetheless data-driven recovery of compact, semantically rich facade models cannot be ignored - given that it is a vital component of ACCRA - the automatic temporal update agent.

In terms of the immediate future - the key check-point for the airborne massing reconstruction is matching the geometric performance of human CAD technicians in terms of model-quality and compactness. In this sense, this research has certainly moved us closer to this goal and to an extent identified the constituent components that are required. Although one could argue that the data-driven procedural approach has certain vital benefits over the existing approaches, there are still aspects that limit its performance. This thesis documented the further investigations that seek to address this. Essentially, although this research has demonstrated the feasibility of efficient, accurate high-quality architectural asset creation driven procedurally rather than via stochastic methods, practically the implementations discussed represent a very early stage of development - and further research is required to iron out the kinks in the currently employed ideologies.

Nonetheless, the key take-home message of this thesis is that fast, accurate and sparse architectural reconstruction methods are not only possible, but in fact highly practical. Ultimately it does not make sense to randomly guess, when by simply applying stable intuited insights about the nature of the built-environment, analogous compact 3D building assets can be constructed in a fraction of the time. However, more over (than the inherent limitations of sampling), procedurally driven strategies (that construct semantically-rich representations) provide one crucial advantage that is currently unmatched by the existing algorithms - they act as a basis for the future of architectural reconstruction - the recovery of *smart* BIM models.

...

So ends this foray into the realm of architectural reconstruction.

All that remains is to acknowledge the supporting parties.

6.4 Acknowledgements

Funding

First - much gratitude is due to the various organisations that supported this research financially. This research was funded and sponsored by:

- An Engineering and Physical Science Research Council (EPSRC) doctoral training grant.
- The Centre for Digital Entertainment (CDE) within the Department of Computer Science at the University of Bath.
- The Architectural Visualisation firm Cityscape Digital.

LiDAR and CAD Datasets

Further - the following acknowledgements credit the external sources of the geometric data used during the course of the research:

- Thanks to the Department of Civil Engineering at the University of Bath for access to the City of Bath CAD model.
- Thanks to the Environments Agency (through its subsidiary the Geomatics Group - now environment.data.gov.uk) for open-access to the airborne scan datasets of the cities of Bath, Manchester, and London.
- Thanks to the MSA-Group for access to the Roslyn Mews unstructured ground scan dataset employed in ground reconstruction.
- Thanks to Cityscape-Digital for access to the Tower Bridge unstructured ground-scan dataset employed in multi-modal reconstruction.

Feedback and Review

Thanks to the assorted anonymous reviewers for their helpful feedback to the submissions derived from this research. Additionally special thanks to Paul and Matt for their technical and theoretical insights and feedback throughout the project, Darren and Wassim for their critical analysis of this thesis and Yongliang for his insightful comments regarding the exposition of the airborne reconstruction.

Support

Finally - thank you again to the individuals whose help and support eased the progression of this research. Many many thanks to my fellow Cityscapers, my supervisory team, my family and the entire CDE.

Quick, Unconstrained, Approximate L-Shape Method

K. Edum-Fotwe[1,2], P. Shepherd[1], M. Brown[1], D. Harper[2], R. Dinnis[2]
[1]University of Bath [2]Cityscape Digital

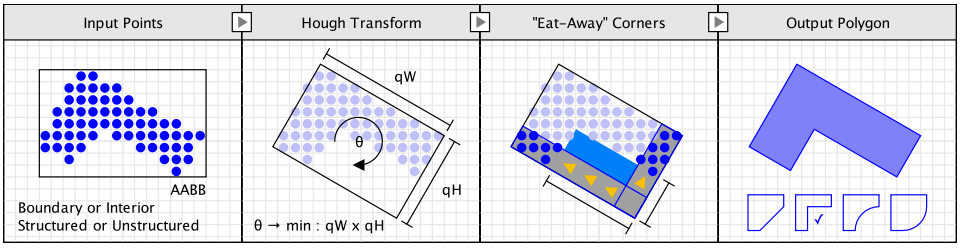

Figure 1: *Overview of the simple 2D shape approximation function - QUALM : (from left to right) the input points, the minimal area bounding box, then reducing error by 'eating-away' corners, and finally the output polygon (with alternative eat-away corner types illustrated below).*

Abstract

This simple paper describes an intuitive data-driven approach to reconstructing architectural building-footprints from structured or unstructured 2D pointsets. The function is fast, accurate and unconstrained. Further unlike the prevalent L-Shape detectors predicated on a shape's skeletal descriptor [Szeliski 2010], the method is robust to sensing noise at the boundary of a 2D pointset.

Keywords: Shape Detection, Hough Transform, *Eat-Away* Hull

Concepts: •**Computing methodologies** → **Shape modeling;**

1 Introduction and Motivation

The context of this work is the automatic recovery of clean (sparse) architectural geometry from various types of laser scan. In particular this operator aims to recover compact building footprints - that can be used for updating 2D-maps and for 3D urban modelling.

The method applies a simple observation about the nature of common rectilinear forms, in order to 'eat-away' at a minimal-area bounding box of a cluster of 2D points. One of the key benefits is determinism. Each 'eat-away' hull represents a repeatable product of the input-points. Another key benefit is resolution independence, since the method does not constrain the point-spacing of the input.

The approach executes in two stages (illustrated in fig.1). First it computes the minimal area bounding box (MABB) of the input 2D points. It then refactors each corner of the MABB by approximating the maximal inset edge-lengths, and injecting a corresponding 'eaten-away' right-angled corner in place of the MABB vertex. The appendix contains the implementation of the technique.

Measuring Geometric Error - Since this is a heuristic shape approximation method, it is vital to be able to measure the accuracy of each generated polygon relative to the input-points. For this two measures are considered. A discrete maximum point-to-edge distance and a continuous normalised shape-to-shape-overlap ratio. They enable an automatic algorithm to quantify the geometric fit.

The Discrete Hausdorff-Distance Error Measure

$$f(A,B) = max(||A_i - (B_j, B_{j+1})||) \;\; \forall i \in A : \forall j \in B$$

The Continuous Intersect-over-Union Error Measure

$$(A \cap B)/(A \cup B) > \omega : \omega \in [0:1]$$

SIGGRAPH '16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4371-8/16/07
DOI: http://dx.doi.org/10.1145/2945078.2945163

2 Results

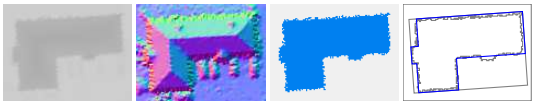

Figure 2: *an example from the 50cm point-spacing London dataset illustrating (from left to right) input-range-points, normals, difference of elevation building segment, resulting automatic l-shape footprint (scan-converted boundary in gray, eat-away hull in blue)*

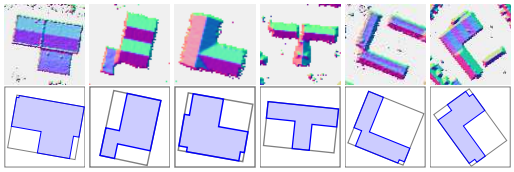

Figure 3: *Building footprints automatically recovered from 1m point-spacing airborne range scans of the city of Bath, UK*

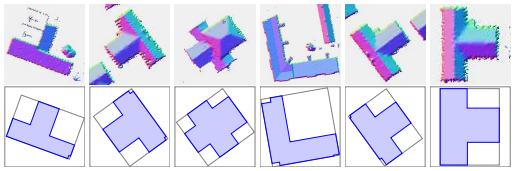

Figure 4: *Building footprints automatically recovered from 25cm point-spacing airborne range scans of the city of Manchester, UK*

References

SZELISKI, R. 2010. *Computer vision: algorithms and applications*. Springer.

Appendix

This page presents the implemented 'eat-away' function - used to automatically recover the building footprints illustrated in the results section.

function QUALM ($points,\ hull,\ min\_dist$) $\rightarrow$ ***Q***uick ***U***nconstrained ***A***pproximate ***L***-Shape ***M***ethod

$points$ - a set of unstructured or structured 2D points
$hull$ - an optional dense extremal boundary hull for the input pointset (to speed up the hough-transform)
min_dist - the minimum length of an edge in an eat-away-corner (a positive scalar to control the minimum inset size)
return value - a 2D polygon : a sequence of vertices representing the detected L-Shape, T-Shape or S-Shape (0-4 refactored corners)

$ret \leftarrow \{\}$
$quad \leftarrow hough\_transform\_minimal\_area\_quad(hull\ ?\ hull : points)$

for $i \leftarrow 0 : i < 4$ **do**
  $min\_distance \leftarrow minimum\_distance\_between\_point\_and\_polygon(quad[i], hull\ ?\ hull : points)$
  if $min\_distance > min\_dist$ **then**
    $prev \leftarrow quad[i > 0\ ?\ i - 1 : 3]$
    $pos \leftarrow quad[i]$
    $next \leftarrow quad[i < 3\ ?\ i + 1 : 0]$

    $prev\_dx \leftarrow pos_x - prev_x$
    $prev\_dy \leftarrow pos_y - prev_y$
    $next\_dx \leftarrow next_x - pos_x$
    $next\_dy \leftarrow next_y - pos_y$
    $prev\_len \leftarrow sqrt(prev\_dx \times prev\_dx + prev\_dy \times prev\_dy)$
    $next\_len \leftarrow sqrt(next\_dx \times next\_dx + next\_dy \times next\_dy)$
    $prev\_ext \leftarrow (prev\_len - min\_distance)/prev\_len$
    $next\_ext \leftarrow min\_distance/next\_len$

    $prev\_half\_quad \leftarrow \{$
      $prev,$
      $pos,$
      $vec2D(pos_x + next\_dx \times next\_ext \times 0.5, pos_y + next\_dy \times next\_ext \times 0.5),$
      $vec2D(prev_x + next\_dx \times next\_ext \times 0.5, prev_y + next\_dy \times next\_ext \times 0.5)$
    $\}$
    $next\_half\_quad \leftarrow \{$
      $pos,$
      $next,$
      $vec2D(next_x - prev\_dx \times (1 - prev\_ext) \times 0.5, next_y - prev\_dy \times (1 - prev\_ext) \times 0.5),$
      $vec2D(pos_x - prev\_dx \times (1 - prev\_ext) \times 0.5, pos_y - prev\_dy \times (1 - prev\_ext) \times 0.5)$
    $\}$

    $prev\_points\_in\_half \leftarrow points\_inside\_polygon(points, prev\_half\_quad)$
    $next\_points\_in\_half \leftarrow points\_inside\_polygon(points, next\_half\_quad)$

    $prev\_min\_distance \leftarrow distance\_to\_closest\_neighbour(pos, prev\_points\_in\_half)$
    $next\_min\_distance \leftarrow distance\_to\_closest\_neighbour(pos, next\_points\_in\_half)$

    if $prev\_min\_distance > next\_min\_distance$ **then**
      $prev\_ext \leftarrow (prev\_len - prev\_min\_distance)/prev\_len$
    else $next\_ext \leftarrow next\_min\_distance/next\_len$
    end if

    $new\_prev \leftarrow vec2D(prev_x + prev\_dx \times prev\_ext, prev_y + prev\_dy \times prev\_ext)$
    $add(new\_prev, ret)$ ▷ new prev
    $add(vec2D(new\_prev_x + next\_dx \times next\_ext, new\_prev_y + next\_dy \times next\_ext), ret)$ ▷ new pos
    $add(vec2D(pos_x + next\_dx \times next\_ext, pos_y + next\_dy \times next\_ext), ret)$ ▷ new next

  else $add(quad[i], ret)$
  end if
  $i++$
end for

return ret
end function

Fast, Accurate and Sparse, Automatic Facade Reconstruction from Unstructured Ground Laser-Scans

K. Edum-Fotwe[1,2], P. Shepherd[1], M. Brown[1], D. Harper[2], R. Dinnis[2]
[1]University of Bath [2]Cityscape Digital

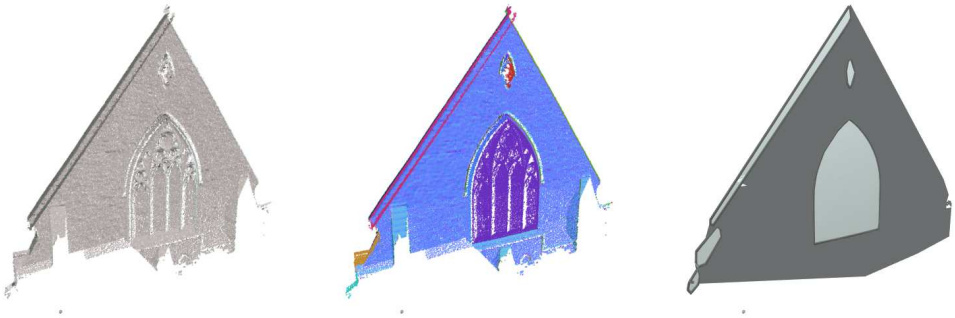

Figure 1: *ARROW → **A**ccurate **R**ailed **R**econstruction of **O**penings and **W**alls : (left) input unstructured points, (middle) signed-distance-field-split with segmented elements, (right) the output polygon-mesh (composed of a quad-dominant wall and railed surface-elements).*

Abstract

This simple paper describes an intuitive data-driven approach to reconstructing architectural facade models from unstructured point-clouds. The algorithm presented yields sparse semantically-rich models that are better suited to interactive simulation than the equivalent dense-reconstructions, yet executes significantly faster than the prevalent sparse-operators. The key advantages include accuracy, efficiency and the ability to model irregular windows.

Keywords: laser scanning, LiDAR, architectural reconstruction, procedural modelling, window detection, pointset segmentation

Concepts: •**Computing methodologies** → **Shape modeling;** *Object detection; Reconstruction;* Point-based models;

1 Introduction and Motivation

The aim this method is to recover compact 3D geometric models of facades that can be used within interactive visualisations and simulations of the real-world. The key limitations of current data-driven facade reconstruction methods are the quality of the generated geometry, and the associated execution times [Musialski et al. 2013]. However methods that exploit model-based templating strategies (although yielding cleaner meshes) are subject to the loss of geometric accuracy and further are constrained by the primitives present in the 'model-library' or 'knowledge-base' [Szeliski 2010]. In such cases it is common to exploit a 2D-split-logic representation of a facade and regularise windows and doors using 'snap-line' techniques [Müller et al. 2007]. Although this often works for Manhattan-style facades, irregular facades (such as the church-face in fig. 1), require data-driven algorithms. The key challenges in facade-reconstruction are segmentation and polygonisation.

SIGGRAPH '16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4371-8/16/07
DOI: http://dx.doi.org/10.1145/2945078.2945123

2 The Algorithm

The section presents the algorithm used to reconstruct the facade in figure 1 and those illustrated in the results. The steps are simple:

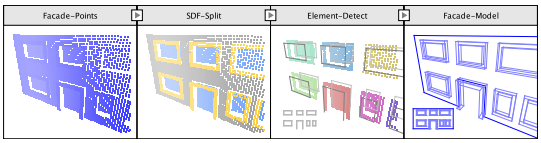

Figure 2: *Key stages in ground facade reconstruction - (left to right): input unstructured point-cloud, signed-distance-field split, segmented surface-element rails, and the resultant facade model.*

Figure 3: *Intermediary data exploited by the ARROW algorithm to produce the result displayed in figure 1 - illustrating (from left to right, top to base) quantized wall-point filter response, binary division resulting from the SDF-split (wall-points in gray, deviant-points in blue), KD-tree used to 'chunk' sets of deviant points, connected component clusters extracted from KD-tree chunks, railed surface-element models, and finally quad-dominant wall model.*

The key idea is to efficiently slice each facade into two sets of points (one containing facade points the other containing salient feature points). Then to cluster each locale of salient points based on connectivity (disjointness). To ensure efficient point-location queries

the operator exploits a simple AABB spatial optimisation and a KD tree space partitioning. Figures 2 and 3 illustrate the key stages and the intermediary data used by the operator. The following pseudo-code seeks to clarify the lower-level behavioural characteristics.

ARROW → ***A***ccurate ***R***ailed ***R***econstruction of ***O***penings & ***W***alls

function RECONSTRUCT($pointset, sdf\_offset$)

$p \leftarrow pointset$
$d \leftarrow sdf\_offset$
if $(p \equiv null | p.points \equiv null | p.points.length = 0)$ **then**
  return $null$;
end if
$sdf\_split \leftarrow signed\_distance\_field\_split(p, d)$
$wall\_points \leftarrow sdf\_split[0]$
$deviant\_points \leftarrow sdf\_split[1]$
$elem\_clusters \leftarrow connected\_comps(deviant\_points)$
$elems \leftarrow \{\}$
for $ptset : elem\_clusters$ **do**
  $se \leftarrow railed\_surface\_element(ptset)$
  if $se \neq null$ **then**
    $elems.add(se)$;
  end if
end for
$wall\_mesh \leftarrow sparse\_cutout\_wall(wall\_points, elems)$
$merged\_elems \leftarrow surface\_elements\_to\_mesh(elems)$
$facade \leftarrow \{wall\_mesh, merged\_elems\}$
return $facade$
end function

Two key insights enable the efficient recovery of sparse-geometry. Foremost the operator uses polygon clipping and 3D-sweeps in order to control the topology of facade models. Further by localising on each facade's wall, the operator can project the 3D points onto a 2D plane to speed up the process of extracting vector profiles.

3 Results

This section presents early qualitative results of the algorithm.

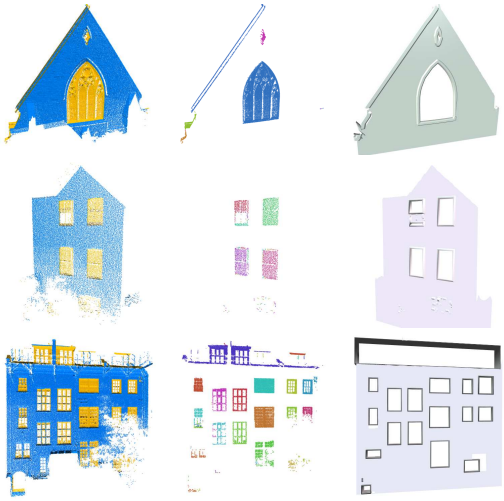

Figure 4: *Facades segmented and reconstructed by the algorithm*

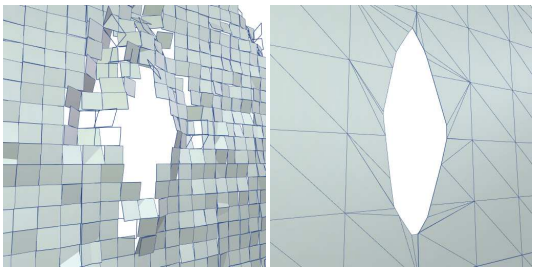

Figure 5: *Comparing the topology of volumetric reconstruction using the regular arrangement of planes (left) to the algorithm (right)*

3.1 Further Research

In terms of the future, there are still vital problems to address to enhance the performance and utility of this reconstruction operator. This final section outlines some of the key further challenges.

Surface-Element Reconstruction - boils down to automatically resolving procedural split logics in order to enable data-driven modelling of segmented window clusters using sets of generalised cylinders. Figure 6 illustrates a formal representation suitable for this.

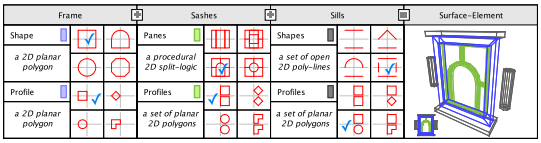

Figure 6: *an abstract formal representation of a window - suitable for data-driven reconstruction and model-based template fitting*

Non-Planar and Curved-Facades present tricky cases for the operator. This is largely due to the sensitivity of the binary-split. This is because it requires a precise wall-representation which is more expensive to recover and manipulate in the case of curved facades.

Multi-Modal Reconstruction: Merging Aerial & Ground Scans would enable the automatic reconstruction (and temporal-update) of complete (mass and surface-detail) *sparse* city-scale models. This is the current focus of continued research - figure 7 illustrates.

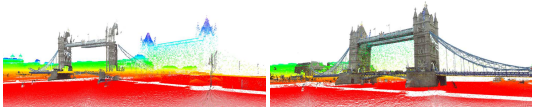

Figure 7: *further investigations - automatically merging multiple sources of laser scan for complete building reconstruction*

References

MÜLLER, P., ZENG, G., WONKA, P., AND VAN GOOL, L. 2007. Image-based procedural modeling of facades. In *ACM Transactions on Graphics (TOG)*, vol. 26, ACM, 85.

MUSIALSKI, P., WONKA, P., ALIAGA, D. G., WIMMER, M., GOOL, L., AND PURGATHOFER, W. 2013. A survey of urban reconstruction. In *Computer graphics forum*, vol. 32, Wiley Online Library, 146–177.

SZELISKI, R. 2010. *Computer vision: algorithms and applications*. Springer.

Efficiently Extracting Extrusions from Airborne Range Scans for Animation and Visualisation of Urban Environments

K. Edum-Fotwe[1,2], P. Shepherd[1], D. Harper[2], R. Dinnis[2]
[1]University of Bath, [2]Cityscape Digital

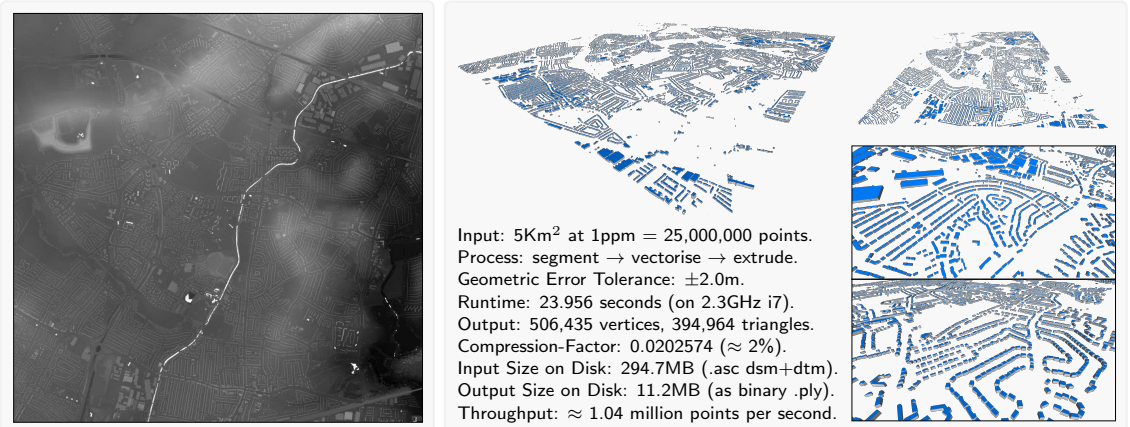

Figure 1: *input (left) and output (right) of the technique discussed - depicting (left) DSM range-scan and (right) results and 3D models.*

2017

Keywords: Architectural Reconstruction, Active Sensing, Point Cloud Processing, Laser-Scanning, Geometry Processing, Image Processing, Segmentation, Boundary Extraction, Shape Simplification, Geographic Information Systems, 3D Urban Mapping

Abstract

This short article discusses an internal research project undertaken at Cityscape-Digital in February of 2017 to determine the feasibility of automatically constructing compact 3D building models from airborne laser-scans for use in rendering and animation projects. Essentially this article outlines an efficient method of reconstructing architectural contextual massing models from off-the-shelf open-access digital elevation models (DEMs), and documents the outcomes of the investigation, noting both the advantages and limitations of the approach.

1 Introduction

This section outlines the aims and objectives of the work.

- Accuracy - Geometric Precision
- Efficiency - Speedy Execution
- Compact - Sparse (Lightweight) Models
- Scalable - Linear-Runtime-Growth
- Reliable - Deterministic (Repeatability)

2 Background and Context

Within AEC (architecture, engineering and construction) 3D point-clouds provide a basis for comparing as built environments to design plans. Further they can be used to create virtual 3D city-models of physical regions. Such point clouds can be acquired through photogrammetry (image-inversion) or laser-scanning (active-depth-sensing). This work relates to active depth sensing. Now although photographic data is more widely available - it is often difficult to quantify the geometric accuracy of photogrammetrically derived models relative to the real-world. On the other hand, whilst laser-scans provide greater accuracy - they can be more computationally expensive to process. For more information see [1] and [2].

3 Implementation

The implementation of the automatic extrusion extractor is incredibly simple and involves the following three operations:

- **Segment** - Difference-of-Elevation-Models.
- **Vectorise** - Scan-Conversion and Polygon Simplification.
- **Extrude** - Linear-Search to determine elevated heights.

4 Discussion

The automatic method is fast, accurate and reasonably sparse. However model quality could be improved. One way to address this is with post-execution human validation (see figure 2).

Figure 2: *Improving model quality with efficient post-execution human validation.*

References

[1] Przemyslaw Musialski, Peter Wonka, Daniel G Aliaga, Michael Wimmer, L Gool, and Werner Purgathofer. A survey of urban reconstruction. In *Computer graphics forum*, volume 32, pages 146–177. Wiley Online Library, 2013.

[2] Qian-Yi Zhou. *3d urban modeling from city-scale aerial lidar data*. University of Southern California, 2012.

References

[1] Ahsan Abdullah, Reema Bajwa, Syed Rizwan Gilani, Zuha Agha, Saeed Boor Boor, Murtaza Taj, and Sohaib Ahmed Khan. 3d architectural modeling: Efficient ransac for n-gonal primitive fitting. In *Eurographics (Short Papers)*, pages 5–8, 2015.

[2] Oswin Aichholzer, Franz Aurenhammer, David Alberts, and Bernd Gärtner. A novel type of skeleton for polygons. In *J. UCS The Journal of Universal Computer Science*, pages 752–761. Springer, 1996.

[3] Amir Akbarzadeh, J-M Frahm, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, Paul Merrell, M Phelps, S Sinha, B Talton, et al. Towards urban 3d reconstruction from video. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 1–8. IEEE, 2006.

[4] Reha Metin Alkan and Gokcen Karsidag. Analysis of the accuracy of terrestrial laser scanning measurements. In *FIG Working Week*, pages 6–12, 2012.

[5] Konstantinos Bacharidis, Froso Sarri, Vasilis Paravolidakis, Lemonia Ragia, and Michalis Zervakis. Fusing georeferenced and stereoscopic image data for 3d building façade reconstruction. *ISPRS International Journal of Geo-Information*, 7(4):151, 2018.

[6] Luigi Barazzetti, Fabio Remondino, and Marco Scaioni. Combined use of photogrammetric and computer vision techniques for fully automated and accurate 3d modeling of terrestrial objects. In *Videometrics, Range Imaging, and Applications X*, volume 7447, page 74470M. International Society for Optics and Photonics, 2009.

[7] David Belton and Derek D Lichti. Classification and segmentation of terrestrial laser scanner point clouds using local variance information. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5):44–49, 2006.

[8] Alexander C Berg, Floraine Grabler, and Jitendra Malik. Parsing images of architectural scenes. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[9] F Bosche and CT Haas. Automated retrieval of 3d cad model objects in construction range images. *Automation in Construction*, 17(4):499–512, 2008.

[10] Claus Brenner. Interactive modelling tools for 3d building reconstruction. 1999.

[11] Claus Brenner. Towards fully automatic generation of city models. *International Archives of Photogrammetry and Remote Sensing*, 33(B3/1; PART 3):84–92, 2000.

[12] Matthew Carlberg, James Andrews, Peiran Gao, and Avideh Zakhor. Fast surface reconstruction and segmentation with ground-based and airborne lidar range data. Technical report, DTIC Document, 2009.

[13] Will Chang and Matthias Zwicker. Range scan registration using reduced deformable models. In *Computer Graphics Forum*, volume 28, pages 447–456. Wiley Online Library, 2009.

[14] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1261–1268. IEEE, 2010.

[15] George Chen and Avideh Zakhor. 2d tree detection in large urban landscapes using aerial lidar data. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 1693–1696. IEEE, 2009.

[16] Liang Cheng, Jianya Gong, Xiaoling Chen, and Peng Han. Building boundary extraction from high resolution imagery and lidar data. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37(PART B3):693–698, 2008.

[17] Liang Cheng, Jianya Gong, Manchun Li, and Yongxue Liu. 3d building model reconstruction from multi-view aerial imagery and lidar data. *Photogrammetric Engineering & Remote Sensing*, 77(2):125–139, 2011.

[18] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 5556–5565. IEEE, 2015.

[19] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. Meshlab: an open-source mesh processing tool. In *Eurographics Italian Chapter Conference*, volume 2008, pages 129–136, 2008.

[20] Paolo Cignoni and Guido Ranzuglia. Meshlab. http://www.meshlab.net/.

[21] Andrea Cohen, Alexander G Schwing, and Marc Pollefeys. Efficient structured parsing of facades using dynamic programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3206–3213, 2014.

[22] Le Corbusier. *Toward an architecture*. Getty Publications, 2007.

[23] Shreyansh Daftry, Christof Hoppe, and Horst Bischof. Building with drones: Accurate 3d facade reconstruction using mavs. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 3487–3494. IEEE, 2015.

[24] Dengxin Dai, Mukta Prasad, Gerhard Schmitt, and Luc Van Gool. Learning domain knowledge for facade labelling. In *European conference on computer vision*, pages 710–723. Springer, 2012.

[25] Dengxin Dai, Hayko Riemenschneider, Gerhard Schmitt, and Luc Van Gool. Example-based facade texture synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1065–1072, 2013.

[26] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20. ACM, 1996.

[27] L Díaz-Vilariño, J Martínez-Sánchez, S Lagüela, J Armesto, and K Khoshelham. Door recognition in cluttered building interiors using imagery and lidar data. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(5):203, 2014.

[28] Lucía Díaz-Vilariño, Kourosh Khoshelham, Joaquín Martínez-Sánchez, and Pedro Arias. 3d modeling of building indoor spaces and closed doors from imagery and point clouds. *Sensors*, 15(2):3491–3512, 2015.

[29] Oxford English Dictionary. Oxford english dictionary online, 2007.

[30] Min Ding, Kristian Lyngbaek, and Avideh Zakhor. Automatic registration of aerial imagery with untextured 3d lidar models. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[31] P Dorninger and C Nothegger. 3d segmentation of unstructured point clouds for building modelling. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 35(3/W49A):191–196, 2007.

[32] Peter Dorninger and Norbert Pfeifer. A comprehensive automated 3d approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds. *Sensors*, 8(11):7323–7343, 2008.

[33] Matt Duckham, Lars Kulik, Mike Worboys, and Antony Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41(10):3224–3236, 2008.

[34] Kwamina Edum-Fotwe, Paul Shepherd, Matthew Brown, Dan Harper, and Richard Dinnis. Fast, accurate and sparse, automatic facade reconstruction from unstructured ground laser-scans. In *ACM SIGGRAPH 2016 Posters*, page 45. ACM, 2016.

[35] Hongchao Fan, Wei Yao, and Qing Fu. Segmentation of sloped roofs from airborne lidar point clouds using ridge-based hierarchical decomposition. *Remote Sensing*, 6(4):3284–3301, 2014.

[36] Lubin Fan and Peter Wonka. A probabilistic model for exteriors of residential buildings. *ACM Transactions on Graphics (TOG)*, 35(5):155, 2016.

[37] Jean-Dominique Favreau, Florent Lafarge, and Adrien Bousseau. Fidelity vs. simplicity: a global approach to line drawing vectorization. *ACM Transactions on Graphics (TOG)*, 35(4):120, 2016.

[38] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004.

[39] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[40] Christian Fruh and Avideh Zakhor. Constructing 3d city models by merging aerial and ground views. *IEEE computer graphics and applications*, 23(6):52–61, 2003.

[41] Christian Früh and Avideh Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60(1):5–24, 2004.

[42] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 80–87. IEEE, 2009.

[43] Michael Garland. *Quadric-based polygonal surface simplification*. PhD thesis, Georgia Institute of Technology, 1999.

[44] Lucile Gimenez, Sylvain Robert, Frédéric Suard, and Khaldoun Zreik. Automatic reconstruction of 3d building models from scanned 2d floor plans. *Automation in Construction*, 63:48–56, 2016.

[45] D Girardeau-Montaut, M Roux, R Marc, and G Thibault. Change detection on points cloud data acquired with a ground laser scanner. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(part 3):W19, 2005.

[46] Aleksey Golovinskiy, Vladimir G Kim, and Thomas Funkhouser. Shape-based recognition of 3d point clouds in urban environments. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2154–2161. IEEE, 2009.

[47] Environments Agency: Geomatics Group. Airborne lidar scan data repository. http://environment.data.gov.uk/ds/survey/#/survey.

[48] E Gulch, H Muller, and T Labe. Integration of automatic processes into semi-automatic building extraction. *International Archives of Photogrammetry and Remote Sensing*, 32(3; SECT 2W5):177–186, 1999.

[49] Norbert Haala and Martin Kada. An update on automatic 3d building reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):570–580, 2010.

[50] Wen Hao, Yinghui Wang, Xiaojuan Ning, Minghua Zhao, Jiulong Zhang, Zhenghao Shi, Xiaopeng Zhang, et al. Automatic building extraction from terrestrial laser scanning data. *Advances in Electrical and Computer Engineering*, 13(3):11–16, 2013.

[51] Bernhard Hohmann, Ulrich Krispel, Sven Havemann, and Dieter Fellner. Cityfit-high-quality urban reconstructions by fitting shape grammars to images and derived textured point clouds. In *Proceedings of the 3rd ISPRS International Workshop 3D-ARCH*, volume 2009, page 3D, 2009.

[52] Yani Ioannou, Babak Taati, Robin Harrap, and Michael Greenspan. Difference of normals as a multi-scale operator in unorganized point clouds. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 501–508. IEEE, 2012.

[53] Yani Andrew Ioannou. *Automatic urban modelling using mobile urban lidar data*. PhD thesis, 2010.

[54] Andreas Jochem, Bernhard Höfle, Martin Rutzinger, and Norbert Pfeifer. Automatic roof plane detection and analysis in airborne lidar point clouds for solar potential assessment. *Sensors*, 9(7):5241–5262, 2009.

[55] Mark W Jones, J Andreas Baerentzen, and Milos Sramek. 3d distance fields: A survey of techniques and applications. *IEEE Transactions on visualization and Computer Graphics*, 12(4):581–599, 2006.

[56] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *ACM transactions on graphics (TOG)*, volume 21, pages 339–346. ACM, 2002.

[57] Mostafa Kabolizade, Hamid Ebadi, and Salman Ahmadi. An improved snake model for automatic extraction of buildings from urban aerial images and lidar data using genetic algorithm. *Paparoditis N., Pierrot-Deseilligny M., Mallet C., Tournaire O.(Eds), IAPRS*, pages 45–49, 2010.

[58] Martin Kada and Laurence McKinley. 3d building reconstruction from lidar based on a cell decomposition approach. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(Part 3):W4, 2009.

[59] George Kelly and Hugh McCabe. Citygen: An interactive system for procedural city generation. In *Fifth International Conference on Game Design and Technology*, pages 8–16, 2007.

[60] Jan Klein and Gabriel Zachmann. Point cloud collision detection. In *Computer Graphics Forum*, volume 23, pages 567–576. Wiley Online Library, 2004.

[61] Barbara Koch, Ursula Heyder, and Holger Weinacker. Detection of individual tree crowns in airborne lidar data. *Photogrammetric Engineering & Remote Sensing*, 72(4):357–363, 2006.

[62] K Kraus and N Pfeifer. Advanced dtm generation from lidar data. *International Archives Of Photogrammetry Remote Sensing And Spatial Information Sciences*, 34(3/W4):23–30, 2001.

[63] Viraj Kulkarni, Rohan Nagesh, and H Wu. Window detection in frontal façades. *Project work at CS294-69 Image Manipulation and Computational Photography. University of Barkley*, 2011.

[64] Florent Lafarge. *Modèles stochastiques pour la reconstruction tridimensionnelle d'environnements urbains*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris, 2007.

[65] Florent Lafarge. *Some contributions to geometric modeling of urban environments*. PhD thesis, University of Nice Sophia Antipolis, 2014.

[66] Florent Lafarge, Xavier Descombes, Josiane Zerubia, and Marc Pierrot-Deseilligny. Automatic building extraction from dems using an object approach and application to the 3d-city modeling. *ISPRS Journal of photogrammetry and remote sensing*, 63(3):365–381, 2008.

[67] Florent Lafarge, Xavier Descombes, Josiane Zerubia, and Marc Pierrot-Deseilligny. Building reconstruction from a single dem. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[68] Florent Lafarge, Xavier Descombes, Josiane Zerubia, and Marc Pierrot-Deseilligny. Structural approach for building reconstruction from a single dsm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):135–147, 2010.

[69] Florent Lafarge, Renaud Keriven, Mathieu Brédif, and Vu Hoang Hiep. Hybrid multi-view reconstruction by jump-diffusion. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 350–357. IEEE, 2010.

[70] Florent Lafarge, Renaud Keriven, Mathieu Brédif, and Hoang-Hiep Vu. A hybrid multiview stereo algorithm for modeling urban scenes. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):5–17, 2013.

[71] Florent Lafarge and Clement Mallet. Building large urban environments from unstructured point data. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1068–1075. IEEE, 2011.

[72] Florent Lafarge and Clément Mallet. Modeling urban landscapes from point clouds: a generic approach. 2011.

[73] Florent Lafarge and Clément Mallet. Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation. *International journal of computer vision*, 99(1):69–85, 2012.

[74] Chungan Lin and Ramakant Nevatia. Building detection and description from a single intensity image. *Computer vision and image understanding*, 72(2):101–121, 1998.

[75] Hui Lin, Jizhou Gao, Yu Zhou, Guiliang Lu, Mao Ye, Chenxi Zhang, Ligang Liu, and Ruigang Yang. Semantic decomposition and reconstruction of residential scenes from lidar data. *ACM Trans. Graph.*, 32(4):66, 2013.

[76] Jinjie Lin, Daniel Cohen-Or, Hao Zhang, Cheng Liang, Andrei Sharf, Oliver Deussen, and Baoquan Chen. *Structure-preserving retargeting of irregular 3D architecture*, volume 30. ACM, 2011.

[77] Joaquín Martínez, Alex Soria-Medina, Pedro Arias, and Alzir Felippe Buffara-Antunes. Automatic processing of terrestrial laser scanning data of building facades. *Automation in Construction*, 22:298–305, 2012.

[78] Andelo Martinovic, Jan Knopp, Hayko Riemenschneider, and Luc Van Gool. 3d all the way: Semantic segmentation of urban scenes from start to end in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2015.

[79] Andelo Martinović, Markus Mathias, Julien Weissenberg, and Luc Van Gool. A three-layered approach to facade parsing. In *European conference on computer vision*, pages 416–429. Springer, 2012.

[80] Andelo Martinovic and Luc Van Gool. Bayesian grammar learning for inverse procedural modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 201–208, 2013.

[81] Bogdan C Matei, Harpreet S Sawhney, Supun Samarasekera, Janet Kim, and Rakesh Kumar. Building segmentation for densely built urban regions using aerial lidar data. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[82] Markus Mathias, Andelo Martinovic, Julien Weissenberg, and Luc Van Gool. Procedural 3d building reconstruction using shape grammars and detectors. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 304–311. IEEE, 2011.

[83] Leena Matikainen, Juha Hyyppä, and Hannu Hyyppä. Automatic detection of buildings from laser scanner data for map updating. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(3/W13):218–224, 2003.

[84] Leena Matikainen, Juha Hyyppä, and Harri Kaartinen. Automatic detection of changes from laser scanner and aerial image data for updating building maps. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci*, 35:434–439, 2004.

[85] Helmut Mayer. Automatic object extraction from aerial imagerya survey focusing on buildings. *Computer vision and image understanding*, 74(2):138–149, 1999.

[86] Andrew Philip McClune. Automatic reconstruction of three-dimensional building models from dense image matching datasets. 2017.

[87] Facundo Mémoli and Guillermo Sapiro. Comparing point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 32–40. ACM, 2004.

[88] Facundo Mémoli and Guillermo Sapiro. A theoretical and computational framework for isometry invariant recognition of point cloud data. *Foundations of Computational Mathematics*, 5(3):313–347, 2005.

[89] Milos Miljanovic, Thomas Eiter, and Uwe Egly. Detection of windows in facades using image processing algorithms. *Indian Journal of Computer Science Engineering*, 3(4):539–547, 2012.

[90] Domen Mongus, Niko Lukač, and Borut Žalik. Ground and building extraction from lidar data based on differential morphological profiles and locally fitted surfaces. *ISPRS Journal of Photogrammetry and Remote Sensing*, 93:145–156, 2014.

[91] Michel Morgan and Klaus Tempfli. Automatic building extraction from airborne laser scanning data. *International Archives of Photogrammetry and Remote Sensing*, 33(B3/2; PART 3):616–623, 2000.

[92] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, and Luc Van Gool. Procedural modeling of buildings. In *Acm Transactions On Graphics (Tog)*, volume 25, pages 614–623. ACM, 2006.

[93] Pascal Müller, Gang Zeng, Peter Wonka, and Luc Van Gool. Image-based procedural modeling of facades. *ACM Transactions on Graphics (TOG)*, 26(3):85, 2007.

[94] Claudio Mura, Oliver Mattausch, Alberto Jaspe Villanueva, Enrico Gobbetti, and Renato Pajarola. Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. *Computers & Graphics*, 44:20–32, 2014.

[95] Przemyslaw Musialski, Peter Wonka, Daniel G Aliaga, Michael Wimmer, L Gool, and Werner Purgathofer. A survey of urban reconstruction. In *Computer graphics forum*, volume 32, pages 146–177. Wiley Online Library, 2013.

[96] Gen Nishida, Ignacio Garcia-Dorado, Daniel G Aliaga, Bedrich Benes, and Adrien Bousseau. Interactive sketching of urban procedural models. *ACM Transactions on Graphics (TOG)*, 35(4):130, 2016.

[97] Sebastian Ochmann, Richard Vock, Raoul Wessel, and Reinhard Klein. Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, 54:94–103, 2016.

[98] K Oda, T Takano, T Doihara, and R Shibasaki. Automatic building extraction and 3-d city modeling from lidar data based on hough transformation. In *Proceedings of XXth ISPRS Congress*, volume 45, 2004.

[99] Yoav IH Parish and Pascal Müller. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 301–308. ACM, 2001.

[100] Hongjoo Park and Samsung Lim. Data fusion of laser scanned data and aerial orthoimagery for digital surface modeling. In *International Workshop on 3D Geoinformation, Seoul, Korea*, 2008.

[101] Viorica Pătrăucean, Iro Armeni, Mohammad Nahangi, Jamie Yeung, Ioannis Brilakis, and Carl Haas. State of research in automatic as-built modelling. *Advanced Engineering Informatics*, 29(2):162–171, 2015.

[102] Mark Pauly. *Point primitives for interactive modeling and processing of 3D geometry*. PhD thesis, University of Kaiserslautern, 2003.

[103] Marc Pollefeys, David Nistér, J-M Frahm, Amir Akbarzadeh, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, S-J Kim, Paul Merrell, et al. Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, 2008.

[104] Charalambos Poullis and Suya You. Automatic reconstruction of cities from remote sensor data. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2775–2782. IEEE, 2009.

[105] Shi Pu and George Vosselman. Building facade reconstruction by fusing terrestrial laser points and images. *Sensors*, 9(6):4525–4542, 2009.

[106] Timo Pylvänäinen, Jérôme Berclaz, Thommen Korah, Varsha Hedau, Mridul Aanjaneya, and Radek Grzeszczuk. 3d city modeling from street-level data for augmented reality applications. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 238–245. IEEE, 2012.

[107] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. Segmentation of point clouds using smoothness constraint. *International archives of photogrammetry, remote sensing and spatial information sciences*, 36(5):248–253, 2006.

[108] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *Computer Vision–ECCV 2008*, pages 500–513, 2008.

[109] Pasi Raumonen, Eric Casella, Mathias Disney, Markku Åkerblom, and Mikko Kaasalainen. Fast automatic method for constructing topologically and geometrically precise tree models from tls data. In *International Conference on Functional-Structural Plant Models*, pages 89–91, 2013.

[110] Hayko Riemenschneider, Ulrich Krispel, Wolfgang Thaller, Michael Donoser, Sven Havemann, Dieter Fellner, and Horst Bischof. Irregular lattices for complex shape grammar facade parsing. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1640–1647. IEEE, 2012.

[111] F Rottensteiner, J Trinder, S Clode, and KKT Kubik. Fusing airborne laser scanner data and aerial imagery for the automatic extraction of buildings in densely built-up areas. In *The International Society for Photogrammetry and Remote Sensing's Twentieth Annual Congress*, volume 35, pages 512–517. ISPRS, 2004.

[112] Franz Rottensteiner and Christian Briese. A new method for building extraction in urban areas from high-resolution lidar data. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/A):295–301, 2002.

[113] Franz Rottensteiner and Christian Briese. *Automatic generation of building models from LIDAR data and the integration of aerial images*. na, 2003.

[114] Franz Rottensteiner and Josef Jansa. Automatic extraction of buildings from lidar data and aerial images. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(4):569–574, 2002.

[115] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1–4. IEEE, 2011.

[116] M Sajadian and H Arefi. A data driven method for building reconstruction from lidar point clouds. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(2):225, 2014.

[117] Aparajithan Sampath and Jie Shan. Building boundary tracing and regularization from airborne lidar point clouds. *Photogrammetric Engineering & Remote Sensing*, 73(7):805–812, 2007.

[118] Aparajithan Sampath and Jie Shan. Building roof segmentation and reconstruction from lidar point clouds using clustering techniques. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37:279–284, 2008.

[119] Aparajithan Sampath and Jie Shan. Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds. *IEEE Transactions on geoscience and remote sensing*, 48(3):1554–1567, 2010.

[120] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007.

[121] Michael Schwarz and Pascal Müller. Advanced procedural modeling of architecture. *ACM Transactions on Graphics (TOG)*, 34(4):107, 2015.

[122] Muhammad Shahzad and Xiao Xiang Zhu. Robust reconstruction of building facades for large areas using spaceborne tomosar point clouds. *IEEE Transactions on Geoscience and Remote Sensing*, 53(2):752–769, 2015.

[123] Paul Shepherd, Kwamina Edum-Fotwe, Matthew Brown, Dan Harper, and Richard Dinnis. Qualm: Quick, unconstrained approximate l-shape method. *SIGGRAPH 2016*, 2016.

[124] Peter Shirley, Michael Ashikhmin, and Steve Marschner. *Fundamentals of computer graphics*. CRC Press, 2009.

[125] Fasahat Ullah Siddiqui, Shyh Wei Teng, Guojun Lu, and Mohammad Awrangjeb. Automatic extraction of buildings in an urban region. In *Proceedings of the 29th International Conference on Image and Vision Computing New Zealand*, pages 178–183. ACM, 2014.

[126] George Sithole and George Vosselman. Automatic structure detection in a point-cloud of an urban landscape. In *Remote Sensing and Data Fusion over Urban Areas, 2003. 2nd GRSS/ISPRS Joint Workshop on*, pages 67–71. IEEE, 2003.

[127] DD Sokolov. Curvature. *Hazewinkel, Michiel, Encyclopedia of Mathematics*, 2001.

[128] Kenichi Sugihara and Yoshitugu Hayashi. Automatic generation of 3d building models with multiple roofs. *Tsinghua Science and Technology*, 13(S1):368–374, 2008.

[129] Shaohui Sun and Carl Salvaggio. Aerial 3d building detection and modeling from airborne lidar point clouds. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 6(3):1440–1449, 2013.

[130] Yao Sun. *3D building reconstruction from spaceborne TomoSAR point cloud*. PhD thesis, Technical University of Munich, 2016.

[131] Junichi Susaki. Knowledge-based modeling of buildings in dense urban areas by combining airborne lidar data and aerial images. *Remote Sensing*, 5(11):5944–5968, 2013.

[132] Ildiko Suveg and George Vosselman. 3d reconstruction of building models. *International archives of photogrammetry and remote sensing*, 33(B2; PART 2):538–545, 2000.

[133] Sohel Syed, Paul Dare, and Simon Jones. Semi-automatic 3d building model generation from lidar and high resolution imagery. *Proceedings of SSC Spatial Intelligence*, 117, 2005.

[134] Richard Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.

[135] Mirela Tanase and Remco C Veltkamp. Polygon decomposition based on the straight line skeleton. In *Proceedings of the nineteenth annual symposium on Computational geometry*, pages 58–67. ACM, 2003.

[136] Pingbo Tang, Daniel Huber, Burcu Akinci, Robert Lipman, and Alan Lytle. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in construction*, 19(7):829–843, 2010.

[137] Fayez Tarsha-Kurdi, Tania Landes, and Pierre Grussenmeyer. Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from lidar data. In *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, volume 36, pages 407–412, 2007.

[138] Fayez Tarsha-Kurdi, Tania Landes, Pierre Grussenmeyer, and Mathieu Koehl. Model-driven and data-driven approaches using lidar data: Analysis and comparison. In *ISPRS Workshop, Photogrammetric Image Analysis (PIA07)*, pages 87–92, 2007.

[139] Olivier Teboul, Loic Simon, Panagiotis Koutsourakis, and Nikos Paragios. Segmentation of building facades using procedural shape priors. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3105–3112. IEEE, 2010.

[140] Thoreau Rory Tooke. *Building energy modelling and mapping using airborne LiDAR*. PhD thesis, University of British Columbia, 2014.

[141] Linh Truong-Hong, Debra F Laefer, Tommy Hinks, and Hamish Carr. Combining an angle criterion with voxelization and the flying voxel method in reconstructing building models from lidar data. *Computer-Aided Civil and Infrastructure Engineering*, 28(2):112–129, 2013.

[142] Marc Van Kreveld, Thijs Van Lankveld, and Remco C Veltkamp. Watertight scenes from urban lidar and planar surfaces. In *Computer Graphics Forum*, volume 32, pages 217–228. Wiley Online Library, 2013.

[143] Carlos A Vanegas, Daniel G Aliaga, and Bedrich Benes. Building reconstruction using manhattan-world grammars. 2010.

[144] George Vosselman, Sander Dijkman, et al. 3d building model reconstruction from point clouds and ground plans. *International archives of photogrammetry remote sensing and spatial information sciences*, 34(3/W4):37–44, 2001.

[145] Jun Wang and Jie Shan. Segmentation of lidar point clouds for building extraction. In *American Society for Photogramm. Remote Sens. Annual Conference, Baltimore, MD*, pages 9–13, 2009.

[146] Lu Wang and Ulrich Neumann. A robust approach for automatic registration of aerial images with untextured aerial lidar data. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2623–2630. IEEE, 2009.

[147] Oliver Wang. *Using Aerial Lidar Data to Segment and Model Buildings*. PhD thesis, University of California, Santa Cruz, 2006.

[148] Oliver Wang, Suresh K Lodha, and David P Helmbold. A bayesian approach to building footprint extraction from aerial lidar data. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 192–199. IEEE, 2006.

[149] Julien Weissenberg, Hayko Riemenschneider, Mukta Prasad, and Luc Van Gool. Is there a procedural logic to architecture? In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 185–192. IEEE, 2013.

[150] Peter Wonka, Michael Wimmer, François Sillion, and William Ribarsky. *Instant architecture*, volume 22. ACM, 2003.

[151] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M Seitz. Schematic surface reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1498–1505. IEEE, 2012.

[152] Fuzhang Wu, Dong-Ming Yan, Weiming Dong, Xiaopeng Zhang, and Peter Wonka. Inverse procedural modeling of facade layouts. *arXiv preprint arXiv:1308.0419*, 2013.

[153] Jianxiong Xiao, Tian Fang, Ping Tan, Peng Zhao, Eyal Ofek, and Long Quan. Image-based façade modeling. In *ACM transactions on graphics (TOG)*, volume 27, page 161. ACM, 2008.

[154] Biao Xiong, S Oude Elberink, and G Vosselman. Building modeling from noisy photogrammetric point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(3):197, 2014.

[155] Jianhua Yan, Keqi Zhang, Chengcui Zhang, Shu-Ching Chen, and Giri Narasimhan. Automatic construction of 3-d building model from airborne lidar data through 2-d snake algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 53(1):3–14, 2015.

[156] Florent Lafarge Yannick Verdie and Pierre Alliez. Lod generation for urban scenes. *ACM Transactions on Graphics, Association for Computing Machinery*, 34 (3):pp.15, 2015.

[157] Suya You, Jinhui Hu, Ulrich Neumann, and Pamela Fox. Urban site modeling from lidar. In *Computational Science and Its ApplicationsICCSA 2003*, pages 579–588. Springer, 2003.

[158] Zhiding Yu, Chunjing Xu, Jianzhuang Liu, Oscar C Au, and Xiaoou Tang. Automatic object segmentation from large scale 3d urban point clouds through manifold embedded mode seeking. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1297–1300. ACM, 2011.

[159] Lukas Zebedin, Joachim Bauer, Konrad Karner, and Horst Bischof. Fusion of feature- and area-based information for urban buildings modeling from aerial imagery. *Computer Vision–ECCV 2008*, pages 873–886, 2008.

[160] Keqi Zhang, Jianhua Yan, and S-C Chen. Automatic construction of building footprints from airborne lidar data. *IEEE Transactions on Geoscience and Remote Sensing*, 44(9):2523–2533, 2006.

[161] Yuanfan Zheng, Qihao Weng, and Yaoxing Zheng. A hybrid approach for three-dimensional building reconstruction in indianapolis from lidar data. *Remote Sensing*, 9(4):310, 2017.

[162] Qian-Yi Zhou. *3d urban modeling from city-scale aerial lidar data*. University of Southern California, 2012.

[163] Qian-Yi Zhou and Ulrich Neumann. Fast and extensible building modeling from airborne lidar data. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, page 7. ACM, 2008.

[164] Qian-Yi Zhou and Ulrich Neumann. A streaming framework for seamless building reconstruction from large-scale aerial lidar data. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2759–2766. IEEE, 2009.

[165] Qian-Yi Zhou and Ulrich Neumann. 2.5 d dual contouring: A robust approach to creating building models from aerial lidar point clouds. *Computer Vision–ECCV 2010*, pages 115–128, 2010.

[166] Qian-Yi Zhou and Ulrich Neumann. 2.5 d building modeling with topology control. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2489–2496. IEEE, 2011.

[167] Qian-Yi Zhou and Ulrich Neumann. 2.5 d building modeling by discovering global regularities. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 326–333. IEEE, 2012.

[168] Qianyi Zhou. Homepage: Urban modeling. http://qianyi.info/urban.html.

Appendix

- DEV : Detector for Engineering Variance
- MAMMAL : Maximal-Area Mass Modeling of Airborne LiDAR
- GRAILS : Graph-Railed Aproximate Interior Linear Spine
- QUALM : Quick Unconstrained Approximate L-Shape Method
- MARS : Maximal Area Roofshape Segmentation
- ARROW : Accurate Railed Reconstruction of Openings and Walls

DEV Algorithm

Detector for **E**ngineering **V**ariance

This section presents the implementation of the semantic change detector's variance classifier in an abstract programming language akin to Java.

```
1:  function ClassifyUpdateOperator(
        Geometry a, Geometry b,
        double[] t, double dmax, double emax)
2:
3:      /* check input and non-parametric operations */
4:      if (a == null && b == null) return -1;
5:      if (a == null) return 0;   // construct
6:      if (b == null) return 1;   // remove
7:
8:      /* volumetric object representations constructed
9:         from each object's original cluster pointset */
10:     Volume old_vol = new Volume(a.pts);
11:     Volume new_vol = new Volume(b.pts);
12:
13:     /* set operator geometries used by
14:        the pairwise deviance predicates */
15:
16:     /* the exclusive disjunction of the old and new */
```

```
17:     Volume xor = new Volume(old_vol);
18:       xor.exclusiveOr(new_vol);
19:     /* the intersection of the old and new */
20:     Volume its = new Volume(old_vol);
21:       its.intersect(new_vol);
22:     /* the union of the old and new */
23:     Volume uni = new Volume(old_vol);
24:       uni.add(new_vol);
25:
26:     /* check if the objects are already congruent */
27:     double er = meanError(a.pts, b.pts);
28:     double nm = its.mass()/uni.mass();
29:     boolean equiv = (er < dmax && nm < 1 - emax);
30:     if (equiv) return 2;   // congruent
31:
32:     /* determine error for the transform operation */
33:     double err = meanError(transform(a.pts, t), b.pts);
34:     if (err < dmax && err < er) return 3;   // position
35:
36:     /* check the ratio of congruence to deviance -
37:        if deviance dominates automatically replace */
38:     double xa = xor.mass();
39:     double ia = its.mass();
40:     if (xa/ia ≥ 1) return 6;   // replace
41:
42:     /* exclusive disjunction is less than the intersection -
43:        check extent to which the change is characterised
44:        by either increase or decrease (monotype measure) */
45:
46:     /* the reduction regions: old - new */
47:     Volume neg = new Volume(old_vol);
48:       neg.subtract(new_vol);
49:     /* the extension regions: new - old */
50:     Volume pos = new Volume(new_vol);
51:       pos.subtract(old_vol);
52:
53:     /* if the mass decreasing contribution to deviance
54:        is less than emax - then extend the old object */
55:     double na = neg.mass();
56:     if (na/xa ≤ emax) return 4;   // extend
57:
58:     /* if the mass increasing contribution to deviance
59:        is less than emax - then reduce the old object */
60:     double pa = pos.mass();
61:     if (pa/xa ≤ emax) return 5;   // reduce
62:
63:     /* unclassifiyable deviances must be replaced */
64:     return 6;   // replace
65: end function
```

MAMMAL Algorithm

Maximal **A**rea **M**ass **M**odel **A**irborne **L**iDAR

1: **function** RECONSTRUCT(
$dsm, dtm, maxfoot\_err, maxroof\_err, min\_iou, min\_bh, foot\_ar, roof\_ar, max\_var, sc$)

dsm - a digital elevation model range image of the surface of the region to be reconstructed
dtm - a digital elevation model range image of the terrain of the region to be reconstructed
$maxfoot\_err$ - max footprint simplification error tolerance between dense and sparse vectors in meters
$maxroof\_err$ - max roofshape simplification error tolerance between dense and sparse vectors in meters
min_iou - the minimum intersection over union ratio for the interior shape overlap measure in the range [0:1]
min_bh - the minimum height of building in meters : for DoEM segmentation
$foot\_ar$ - the minimum footprint surface-area of a building in meters^2 : for DoEM segmentation
$roof\_ar$ - the minimum desired roofshape surface-area in meters^2 : used by MARS
max_var - the maximum local variance between points belonging to the same region : used by MARS
sc - the model selection criteria to use in ranking of candidate masses : an unsigned scalar
return value - a set of mass-models representing the set of buildings extracted from the input dsm

```
2:    /* segment the buildings in the input dsm using the difference of elevations */
3:    doem ← dsm − dtm
4:    sliced ← threshold(doem, min_bh)
5:    cc ← connected_components(sliced, dsm_w, dsm_h)
6:    cc ← filter(cc, foot_ar, max_var)
7:    /* for each segmented building - maximal area segment and model it's mass */
8:    models ← {}
9:    for all c : cc do
10:       densefoot ← linear_boundary_edge_traverse(c)
11:       sparsefoot ← simplify(densefoot, maxfoot_err)
12:       segments ← max_area_segment(c, roof_ar, max_var)
13:       denseroofs ← linear_internal_edge_traverse(segments)
14:       sparseroofs ← {}
15:       if ensure_watertight then
16:          sparseroofs ← graph_refine(denseroofs, maxroof_err)
17:       else
18:          for all p : denseroofs do
19:             rs ← simplify(p, maxroof_err)
20:             add(rs, sparseroofs)
21:          end for
22:       end if
23:       project = project_2D_3D(sparseroofs, c, maxroof_err)
24:       errors = depth_buffer_XZ(project, c)
25:       if allow_parametrics then /* deterministically optimise the building's mass-model */
26:          param ← optimise(c, sparsefoot, sparseroofs, maxfoot_err, maxroof_err, min_iou)
27:          paramerror ← depth_buffer_XZ(param, c)
28:          if abs_err(paramerror) ≤ maxroof_err then
29:             add(param, models)
30:          else add(project, models)
31:          end if
32:       else add(project, models)
33:       end if
34:    end for
35:    return models
36: end function
```

GRAILS Algorithm

Graph **R**efined(/**R**ailed) **A**pproximate **I**nterior **L**inear **S**pine

---

1: **function** SPINE ($polygon, resolution\_step$)

$polygon$ - an input 2D vertex-list of a simple polygon
$resolution\_step$ - the discretisation step for the MAT
return value - a 2D vertex-list of an open polyline spine

2: $keypoints \leftarrow medial\_axis\_critical\_points($
$polygon, resolution\_step)$
3: $path \leftarrow maximum\_length\_graph\_path(keypoints)$
4: **return** $path$
5: **end function**

1: **function** MEDIAL_AXIS_CRITICAL_POINTS (
$polygon, resolution\_step$)

$polygon$ - an input 2D vertex-list of a simple polygon
$resolution\_step$ - the discretisation step for the MAT
return value - a set of objects of type CritcalPoint

2: $shape \leftarrow polygon$
3: $step \leftarrow resolution\_step$
4: $bounds \leftarrow bounding\_box(shape)$
5: $bx \leftarrow bounds_x, by \leftarrow bounds_y$
6: $bw \leftarrow bounds_{width}, bh \leftarrow bounds_{height}$
7: $icells \leftarrow int(bw/step) + 2$
8: $jcells \leftarrow int(bh/step) + 2$
9: $grid \leftarrow int[icells \times jcells]$
10: $jj \leftarrow 0$
11: **for** $j \leftarrow by : j < by + bh$ **do**
12: $ii \leftarrow 0$
13: **for** $i \leftarrow bx : i < bx + bw$ **do**
14: $p \leftarrow vec2D(i, j)$
15: **if** $polygon\_contains(p_x, p_y, shape)$ **then**
16: $closest\_edge\_id \leftarrow index\_of\_polygon\_...$
$...edge\_closest\_to\_point(shape, p)$
17: $grid[ii + jj \times icells] \leftarrow (closest\_edge\_id + 1)$
18: **end if**
19: $i \leftarrow (i + step)$
20: $ii++$
21: **end for**
22: $j \leftarrow (j + step)$
23: $jj++$
24: **end for**
25: $keypoints \leftarrow vector < CriticalPoint > \{\}$
26: **for** $j \leftarrow 0 : j < jcells$ **do**
27: $nj \leftarrow (j + 1 < jcells)$
28: **for** $i \leftarrow 0 : i < icells$ **do**
29: $ni \leftarrow (i + 1 < icells)$
30: $v \leftarrow grid[i + j \times icells]$
31: $ids \leftarrow \{$
32: $grid[i + j \times icells],$
33: $ni\ ?\ grid[i + 1 + j \times icells] : -1,$
34: $nj\ ?\ grid[i + (j + 1) \times icells] : -1,$
35: $(ni : nj)\ ?\ grid[i + 1 + (j + 1) \times icells] : -1$
36: $\}$
37: **if** $count\_unique\_over\_zero(ids) \geq 3$ **then**
38: $point \leftarrow vec2D(bx + (i + 1) \times$
$step, by + (j + 1) \times step)$
39: $edge\_distance \leftarrow minimum\_distance\_...$
$...between\_point\_and\_polygon(point, shape)$
40: $cp \leftarrow CriticalPoint(point, ids, edge\_distance)$
41: $add(cp, keypoints)$
42: **end if**
43: $i++$
44: **end for**
45: $j++$
46: **end for**
47: $keys \leftarrow keypoints$
48: /* compute sibling relationships for all CriticalPoints */
49: **for** $e \leftarrow 0 : e < |keys|$ **do**
50: **for** $c \leftarrow 0 : c < |keys|$ **do**
51: **if** $e \neq c : keys[e] \rightarrow is\_sibling(keys[c])$ **then**
52: /* print sibling detected debug message */
53: **end if**
54: $c++$
55: **end for**
56: $e++$
57: **end for**
58: **return** $keypoints$
59: **end function**

1: **function** MAXIMUM_LENGTH_GRAPH_PATH ($keypoints$)

$keypoints$ - a set of objects of type CriticalPoint
return value - a 2D vertex-list of an open polyline spine

2: $max\_length \leftarrow 0$
3: $max\_ids \leftarrow null$
4: $temp \leftarrow 0$
5: **for** $i \leftarrow 0 : i < |keypoints|$ **do**
6: $keypoints[i].id \leftarrow i$
7: $keypoints[i] \rightarrow compute\_sibling\_distances()$
8: $i++$
9: **end for**
10: **for** $i \leftarrow 0 : i < |keypoints|$ **do**
11: **for** $c \leftarrow 0 : c < |keypoints|$ **do**
12: **if** $i \neq c$ **then**
13: $path\_ids = shortest\_graph\_path\_between($
$i, c, keypoints)$
14: $temp \leftarrow length\_of\_graph\_path($
$path\_ids, keypoints)$
15: **if** $temp > max\_length$ **then**
16: $max\_length \leftarrow temp$
17: $max\_ids \leftarrow path\_ids$
18: **end if**
19: **end if**
20: $c++$
21: **end for**
22: $i++$
23: **end for**
24: **if** $max\_ids \neq null$ **then**
25: $ret \leftarrow vec2D[|max\_ids|]$
26: **for** $i \leftarrow 0 : i < |ret|$ **do**
27: $ret[i] \leftarrow keypoints[max\_ids[i]].point$
28: $i++$
29: **end for**
30: **return** ret
31: **end if**
32: **return** $null$
33: **end function**

1: **function** LENGTH_OF_GRAPH_PATH ($path\_ids, keypoints$)

$path\_ids$ - a set of integer indexes into the keypoints set
$keypoints$ - a set containing objects of type CriticalPoint
return value -

2: $sum \leftarrow 0$
3: **for** $i \leftarrow 0 : i < |path\_ids| - 1$ **do**
4: $p \leftarrow keypoints[path\_ids[i]]$
5: $n \leftarrow keypoints[path\_ids[i + 1]]$
6: $d \leftarrow (p \rightarrow distance\_to\_sibling\_with\_ID(n.id))$
7: $sum \leftarrow (sum + d)$
8: $i++$
9: **end for**
10: **return** sum
11: **end function**

1: **function** SHORTEST_GRAPH_PATH_BETWEEN (
$aid, bid, keypoints$)

---

aid - an integer index of the starting CriticalPoint
bid - an integer index of the end CritialPoint
$keypoints$ - a set containing objects of type CriticalPoint
return value - a sequence of integer indexes into keypoints

2: **if** $|keypoints| \geq 3$ **then**
3: $len \leftarrow |keypoints|$

```
        for i ← 0 : i < len do
            keypoints[i].id ← i
            i++
        end for
        to_visit ← vector < Node > {}
        ret ← vector < int > {}
        for i ← 0 : i < len do
            add(nodes[i], to_visit)
        end for
        node ← nodes[aid]
        node.distance_to_source ← 0
        node.parent ← aid
        while ! is_empty(to_visit) do
            u ← closest_to_source(to_visit)
            remove(u, to_visit)
            for i ← 0 : i < |u.neighbours| do
                v ← nodes[u.neighbours[i]]
                d ← u.distance_to_source +
                (u → distance_to_neighbour_with_ID(v.id))
                if d < v.distance_to_source then
                    v.distance_to_source ← d
                    v.parent ← u.id
                end if
                i++
            end for
        end while
        node ← nodes[bid]
        while node.id ≠ aid : node.parent ≥ 0 do
            add(node.id, ret)
            node = nodes[node.parent]
        end while
        add(aid, ret)
        return reverse(ret)
    else if |keypoints| == 2) then return int[] {0, 1}
    else if |keypoints| == 1) then return int[] {0}
    else return int[] {}
    end if
end function
```

```
function CLOSEST_TO_SOURCE ( nodes )
    nodes - a set of objects of type Node
    return value - a reference to an object of type Node
    closest_distance ← ∞
    closest_id ← −1
    temp ← 0
    for i ← 0 : i < |nodes| do
        temp ← nodes[i].distance_to_source
        if temp ≤ closest_distance then
            closest_id ← i
            closest_distance ← temp
        end if
        i++
    end for
    return nodes[closest_id]
end function
```

```
function COUNT_UNIQUE_OVER_ZERO ( vals )
    vals - a set of integer values
    return value - count of unique values greater than zero
    temp ← vector < int > {}
    for i ← 0 : i < |vals| do
        if vals[i] > 0 : !contains(temp, vals[i]) then
            add(vals[i], temp)
        end if
        i++
    end for
    return |temp|
end function
```

```
function INDEX_OF_POLYGON_EDGE_CLOSEST_TO_POINT (
    polygon, point )
    polygon - a 2D vertex-list of a simple polygon
    point - a 2D vertex
    return value -
    min_distance ← ∞
    min_id ← −1
    len ← |polygon|
    for i ← 0 : i < len do
        p1 ← polygon[i]
        p2 ← polygon[(i + 1)%len]
        d ← distance_between_line_and_point(
            p1_x, p1_y, p2_x, p2_y, point_x, point_y)
        if d ≤ min_distance then
            min_id ← i
            min_distance ← d
        end if
        i++
    end for
end function
```

```
function MIN_DISTANCE_BETWEEN_POINT_AND_POLYGON (
    point, polygon )
    point - a 2D vertex
    polygon - a 2D vertex-list of a simple polygon
    return value - an positive scalar distance
    p ← point
    len ← |polygon|
    min_distance ← ∞
    pos, next, distance
    for i ← 0 : i < len do
        pos ← polygon[i]
        next ← polygon[(i + 1)%len]
        distance ← distance_between_line_and_point(
            pos_x, pos_y, next_x, next_y, p_x, p_y)
        if distance < min_distance then
            min_distance ← distance
        end if
        i++
    end for
    return min_distance
end function
```

```
function DISTANCE_BETWEEN_LINE_AND_POINT (
    vx, vy, wx, wy, px, py )
    vx, vy - line start x, line start y
    wx, wy - line end x, line end y
    px, py - point x, point y
    return value -
    len1 ← sqrt((vx−wx)×(vx−wx)+(vy−wy)×(vy−wy))
    l2 ← len1 × len1
    if l2 == 0 then return
        sqrt((px − vx) × (px − vx) + (py − vy) × (py − vy))
    end if
    t ← ((px − vx) × (wx − vx) + (py − vy) × (wy − vy))/l2
    if t < 0.0 then return
        sqrt((px − vx) × (px − vx) + (py − vy) × (py − vy))
    else if t > 1.0 then return
        sqrt((px − wx) × (px − wx) + (py − wy) × (py − wy))
    end if
    rx ← vx + t × (wx − vx) /* extent position x */
    ry ← vy + t × (wy − vy) /* extent position y */
    return sqrt((px − rx) × (px − rx) + (py − ry) × (py − ry))
end function
```

```
CriticalPoint
    state - /* the variables declared in each CriticalPoint */
    point : vec2D
    ids : int[]
    distance : double
    id ← −1 : int
    siblings ← {} : vector < CriticalPoint >
    sibling_distances : double[]
    operations - /* the set of functions defined for each instantiated CriticalPoint - each is an instance function (such that it operates on the object reference on which it is invoked) */
```

```
8: CriticalPoint() {}                                          ▷ default-constructor
9: CriticalPoint(p, i, d) {                                    ▷ key-constructor
     point ← p, ids ← i, distance ← d
   }
10: is_sibling(b) {                                            ▷ returns boolean
     if siblings → contains(b) then return true end if
     count ← 0, encountered ← vector < int > {}
     for i ← 0 : i < |ids| do for c ← 0 : c < |b.ids| do
       if ids[i] == b.ids[c] : !encountered.contains(ids[i])
         then count + +, add(ids[i], encountered) end if
     c + + end for i + + end for
     if count ≥ 2 then
       if !(siblings → contains(b)) then
         add(b, siblings) end if
       if !(b.siblings → contains(this)) then
         add(this, b.siblings) end if
       return true
     end if
     return false
   }
11: compute_sibling_distances() {                              ▷ returns void
     sibling_distances ← double[|siblings|]
     for i ← 0 : i < |siblings| do
       sibling_distances[i] ← ||(point − siblings[i].point)||
     i + + end for
   }
12: distance_to_sibling_width_ID(sid) {                        ▷ returns double
     for i ← 0 : i < |sibling_distances| do
       if sid == siblings[i].id then
         return sibling_distances[i] end if
     i + + end for
     return − 1 /* it is not a sibling */
   }
```

```
1: Node

   state - /* the set of variables declared within each Node */

2: id ← −1 : int
3: parent ← −1 : int
4: distance_to_source ← −1 : double
5: visited ← false : boolean
6: neighbours ← {} : vector < int >
7: neighbour_distances : double[]

   operations - /* the set of functions defined for each instantiated Node
   - each is an instance function (such that it operates on the object
   reference from which it is invoked) */

8: Node() {}                                                   ▷ default-constructor
9: Node(id_in, source) {                                       ▷ key-constructor
     id ← id_in
     for i ← 0 : i < |source.siblings| do
       add(source.siblings[i].id, neighbours)
       i + + end for
     neighbour_distances ← source.sibling_distances
   }
10: distance_to_neighbour_with_ID(nid) {                       ▷ returns double
     for i ← 0 : i < |neighbours| do
       if nid == neighbours[i] then
         return neighbour_distances[i] end if
     i + + end for
     return − 1 /* it is not a neighbour */
   }
```

QUALM Function

Quick **U**nconstrained **A**pproximate **L**-Shape **M**ethod

```
1: function APPROXIMATE ( points, hull, min_dist )
```

points - a set of unstructured or structured 2D points
hull - an optional dense extremal boundary hull for the input pointset (to speed up the hough-transform)
min_dist - minimum length of an edge in an eat-away-corner (a positive scalar to control the min inset size)
return value - a 2D polygon : a sequence of vertices representing the detected L-Shape, T-Shape or S-Shape

```
2:    ret ← {}
3:    quad ← hough_transform_minimal_area_quad(hull ? hull : points)
4:
5:    for i ← 0 : i < 4 do
6:       min_distance ← min_distance_between_point_and_polygon(quad[i], hull ? hull : points)
7:       if min_distance > min_dist then
8:          prev ← quad[i > 0 ? i − 1 : 3]
9:          pos ← quad[i]
10:         next ← quad[i < 3 ? i + 1 : 0]
11:
12:         prev_dx ← pos_x − prev_x
13:         prev_dy ← pos_y − prev_y
14:         next_dx ← next_x − pos_x
15:         next_dy ← next_y − pos_y
16:         prev_len ← sqrt(prev_dx × prev_dx + prev_dy × prev_dy)
17:         next_len ← sqrt(next_dx × next_dx + next_dy × next_dy)
18:         prev_ext ← (prev_len − min_distance)/prev_len
19:         next_ext ← (next_len − min_distance)/next_len
20:
21:         prev_half_quad ← {
22:            prev,
23:            pos,
24:            vec2D(pos_x + next_dx × next_ext × 0.5, pos_y + next_dy × next_ext × 0.5),
25:            vec2D(prev_x + next_dx × next_ext × 0.5, prev_y + next_dy × next_ext × 0.5)
26:         }
27:         next_half_quad ← {
28:            pos,
29:            next,
30:            vec2D(next_x − prev_dx × (1 − prev_ext) × 0.5, next_y − prev_dy × (1 − prev_ext) × 0.5),
31:            vec2D(pos_x − prev_dx × (1 − prev_ext) × 0.5, pos_y − prev_dy × (1 − prev_ext) × 0.5)
32:         }
33:
34:         prev_points_in_half = points_inside_polygon(points, prev_half_quad)
35:         next_points_in_half = points_inside_polygon(points, next_half_quad)
36:
37:         prev_min_distance = distance_to_closest_neighbour(pos, prev_points_in_half)
38:         next_min_distance = distance_to_closest_neighbour(pos, next_points_in_half)
39:
40:         if prev_min_distance > next_min_distance then
41:            prev_ext ← (prev_len − prev_min_distance)/prev_len
42:         else next_ext ← next_min_distance/next_len
43:         end if
44:
45:         new_prev ← vec2D(prev_x + prev_dx × prev_ext, prev_y + prev_dy × prev_ext)
46:         /* add new-prev, new-pos and new-next */
47:         add(new_prev, ret)
48:         add(vec2D(new_prev_x + next_dx × next_ext, new_prev_y + next_dy × next_ext), ret)
49:         add(vec2D(pos_x + next_dx × next_ext, pos_y + next_dy × next_ext), ret)
50:
51:      else add(quad[i], ret)
52:      end if
53:      i + +
54:   end for
55:
56:   return ret
57: end function
```

MARS Function

Maximal **A**rea **R**oofshape **S**egmentation

1: **function** MAXIMAL_AREA_ROOFSHAPE_SEGMENT (

dsm, a digital-surface-model airborne range scan
dtm, a digital-terrain-model airborne range scan
min_bh, the minimum height of a building in meters
min_fa, the minimum footprint area in m^2
min_ra, the minimum mean roofshape area in m^2
max_dist, the maximum mean point-to-plane distance for a stable roofshape component
max_local_var, the max local variance between neighbouring points within a roofshape
nc_iter the maximum number of iterations for which the algorithm will 'cancel-noise' by suppressing small non-conformal roofshapes

return value: an integer grid with the same dimensions as the input DEMs - containing labelled IDs for the roofshapes in each building - such that IDs greater than zero map (sequentially) to individual roofshapes

2: $doem \leftarrow dsm - dtm$
3: $mask \leftarrow threshold(doem, min\_bh)$
4: $cc \leftarrow connected\_component(mask, dsm_W, dsm_H)$
5: $cc \leftarrow filter\_gt(cc, min\_fa)$
6: $elems \leftarrow clusters(cc, dsm, dtm, doem)$
7: $ret \leftarrow \{\}$
8: $print('connected\ components:\ ' + str(|elems|))$
9: **for** $i \leftarrow 0 : i < |elems|$ **do**
10: $rs \leftarrow label\_graph\_regions(elems_i, max\_dist, max\_local\_var,)$
11: $rmax \leftarrow maximise\_areas(rs, min\_ra)$
12: $rmean \leftarrow mean\_area(rs)$
13: $s \leftarrow 0$
14: **while** $rmean < min\_ra : s < nc\_iter$ **do**
15: $rmax \leftarrow suppress\_non\_conformals(rmax, min\_ra, elems_i)$
16: $rmean \leftarrow mean\_area(rmax)$
17: $s++$
18: **end while**
19: $add(rmax, ret)$
20: **end for**
21: **return** $mask\_from\_segments(dsm_W, dsm_H, ret)$
22: **end function**

ARROW Algorithm

Accurate **R**ailed **R**econstruction of **O**penings & **W**alls

```
1: function RECONSTRUCT(pointset, sdf_offset)
2:     p ← pointset
3:     d ← sdf_offset
4:     if (p ≡ null|p.points ≡ null|p.points.length = 0) then
5:         return null;
6:     end if
7:     sdf_split ← signed_distance_field_split(p, d)
8:     wall_points ← sdf_split[0]
9:     deviant_points ← sdf_split[1]
10:    elem_clusters ← connected_comps(deviant_points)
11:    elems ← {}
12:    for ptset : elem_clusters do
13:        se ← railed_surface_element(ptset)
14:        if se ≠ null then
15:            elems.add(se);
16:        end if
17:    end for
18:    wall_mesh ← sparse_cutout_wall(wall_points, elems)
19:    merged_elems ← surface_elements_to_mesh(elems)
20:    facade ← {wall_mesh, merged_elems}
21:    return facade
22: end function
```